Developpez Le Mag

Édition d'octobre – novembre 2014
Numéro 54
Magazine en ligne gratuit
Diffusion de copies conformes à l'original autorisée
Réalisation : Alexandre Pottiez
Rédaction : la rédaction de Developpez
Contact : magazine@redaction-developpez.com

Sommaire

Éditorial

Voici un nouveau numéro de votre magazine préféré, à lire et à relire à l'infini pour votre plus grand plaisir !

La rédaction

Article Oracle

Optimisation des LDD

Cet article va expliquer la technique d'optimisation LDD, intégrée à Oracle depuis la version 11g.
par Mohamed Houri
Page 2

Article JavaScript

Débat : Les Web Components, pour le meilleur et pour le pire ?

Les Web Components ont fait beaucoup parler d'eux depuis l'avancée des dernières spécifications et le développement de *polyfills* permettant de les utiliser dès maintenant. Ils ont même été décrits par plusieurs articles de presse comme la prochaine révolution du développement web. Mais cet engouement est-il justifié ?
par Sylvain Pollet-Villard
Page 29

Oracle

Les derniers tutoriels et articles

Optimisation des LDD

Cet article va expliquer la technique d'optimisation LDD, intégrée à Oracle depuis la version 11g.

1 Introduction

À partir de sa version 11g, Oracle a introduit une technique très performante qui permet l'optimisation des opérations d'ajout de colonnes sur une table. Lorsque vous ajoutez à une table une colonne « **not null** » et qu'en même temps vous attachez une valeur **par défaut** à cette colonne, alors une nouvelle technique d'optimisation LDD a lieu permettant à cette opération d'être instantanée. Comment cela est-il possible lorsque la table ainsi modifiée possède des millions d'enregistrements qui doivent voir leur nouvelle colonne ajoutée mise à jour avec la valeur par défaut ? Et est-ce que cette nouvelle technique d'optimisation des opérations LDD a été implémentée sans aucun effet secondaire que nous devrions connaître avant son utilisation ? C'est ce que je vais vous expliquer dans cet article.

2 Le concept

Considérons la table suivante contenant trois millions d'enregistrements :

```
1
2  SQL> create table t1
3
4       as select
5          rownum n1
6        , trunc ((rownum-1)/3) n2
7        , trunc(dbms_random.value(rownum,
             rownum*10)) n3
8        , dbms_random.string('U', 10) c1
9       from dual
10      connect by level <= 3e6;
```

```
1  SQL> desc t1
2
3              Name                  Null? Type
4              ----------------------- -----
                   ----
5       1        N1
             NUMBER
6       2        N2
             NUMBER
7       3        N3
             NUMBER
8       4        C1
             VARCHAR2(4000 CHAR)
```

Table à laquelle je vais ajouter une colonne supplémentaire non nulle et ayant une valeur par défaut. Quelque chose comme ceci :

```
1  SQL> alter table t1 add C_DDL number
       default 42 not null;
```

où j'ai mis en gras les deux plus importants mots à savoir **default** et **not null** parce qu'ils représentent les clauses qui gèrent cette nouveauté.

Pour mieux apprécier la différence dans le temps d'exécution de l'instruction 'alter table' ci-dessus, je vais l'exécuter dans deux différentes versions de la base de données Oracle, 10.2.0.4.0 et 11.2.0.3.0 respectivement :

```
1  10.2.0.4.0 > alter table t1 add C_DDL
       number default 42 not null;
2
3  Table altered.
4  Elapsed: 00:00:48.53
```

```
1  11.2.0.3.0> alter table t1 add C_DDL
       number default 42 not null;
2
3  Table altered.
4  Elapsed: 00:00:00.04
```

Observez la différence dans le temps d'exécution. La colonne C_DDL a été ajoutée instantanément dans la version 11gR2 alors que son ajout dans la version 10gR2 a nécessité 49 secondes. Quel est donc ce nouveau mécanisme qui permet cet extrêmement rapide temps d'exécution lors de l'ajout d'une colonne non nulle ayant une valeur par défaut à une table existante ?

Comment trois millions d'enregistrements peuvent-ils être mis à jour en quatre millisecondes ?

Vérifions visuellement si cet update a été réellement fait (à partir de maintenant, lorsque la version

d'Oracle n'est pas précisée, il s'agira alors implicitement de la version 11.0.2.3).

```
SQL> select count(1) from t1;

  COUNT(1)
----------
   3000000
```

```
SQL> select count(1) from t1 where c_ddl
     = 42;

  COUNT(1)
----------
   3000000
```

Bien qu'Oracle ait modifié la table t1 instantanément, la requête montre que la totalité des colonnes C_DDL a été mise à jour avec sa valeur par défaut 42. Comment cela est-il possible ? Est-ce que le plan d'exécution peut nous aider ici ?

```
SQL> select * from table(dbms_xplan.
    display_cursor);
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |  3016 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |     3 |            |          |
|* 2  |   TABLE ACCESS FULL| T1   |  2999K|  8798K|  3016   (5)| 00:00:10 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(NVL("C_DDL",42)=42)
```

Notez bien encore ici comment la partie prédicat du plan d'exécution précédent peut révéler des informations capitales pour la compréhension de ce qui se passe derrière la scène. Je n'ai pas utilisé la fonction NVL dans ma requête, mais elle apparaît quand même dans la partie prédicat indiquant qu'Oracle considère encore que la colonne C_DDL est susceptible de contenir des valeurs nulles (ce qui signifie, qu'en réalité, cette colonne n'a pas été mise à jour, et c'est la raison pour laquelle Oracle est en train de la remplacer par sa valeur par défaut 42).

Nous avons la version précédente d'Oracle à notre disposition pour comprendre et localiser les différences :

```
10.2.0.4.0 > select count(1) from t1
    where c_ddl = 42;

  COUNT(1)
----------
   3000000
```

```
10.2.0.4.0> select * from table(
    dbms_xplan.display_cursor);
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |  4001 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |     3 |            |          |
|* 2  |   TABLE ACCESS FULL| T1   |  3000K|  8789K|  4001   (8)| 00:00:09 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter("C_DDL"=42)
```

L'absence de la fonction NVL dans la partie prédicat couplée au temps qui a été nécessaire pour l'ajout de la colonne C_DDL dans 10gR2 (00 :00 :48.53) expliquent le fonctionnement du concept, introduit en 11gR1, d'optimisation des ajouts de colonnes non nulles ayant une valeur par défaut à une table existante.

À partir de la version 11gR1 d'Oracle, lorsqu'on ajoute une colonne **non nulle** ayant une valeur par **défaut**, Oracle ne va pas mettre à jour tous les enregistrements existants avec cette valeur par défaut. Il va, par contre, stocker une métadonnée pour cette nouvelle colonne (contrainte non nulle et valeur par défaut 42) et va permettre ainsi au LDD d'être accompli instantanément, et ce, quelle qu'ait été la taille de la table modifiée. Bien sûr, cela est possible moyennant l'utilisation de la fonction NVL lors de la lecture de cette nouvelle colonne à partir d'un bloc de la table.

Après avoir expliqué ce magnifique concept d'optimisation LDD, je vais, dans le paragraphe suivant, investiguer un peu plus, sur comment cette nouveauté est gérée par Oracle pour assurer une rapidité de l'opération LDD et pour garantir un résultat correct et performant lors de la sélection des données. Nous allons particulièrement voir que la sélection de la colonne ajoutée à partir d'un bloc de table diffère de celle faite à partir d'un bloc feuille (*leaf block*) d'un index.

3 Effets du mécanisme LDD

3.1 Sur la table modifiée

Nous avons vu plus haut que nous gagnons en performance lors de l'ajout d'une colonne non nulle ayant une valeur par défaut. Par contre, nous avons vu aussi que cela est possible parce qu'Oracle utilise la fonction NVL pour l'appliquer à la nouvelle colonne C_DDL afin de lui attacher sa valeur par défaut. C'est aussi grâce à la métadonnée de la colonne C_DDL qui a été stockée dans le dictionnaire des données. Est-ce que l'utilisation de cette fonction NVL génère un effet secondaire ?

Premièrement nous avons vu précédemment que cela n'a aucune influence sur les estimations faites par le CBO qui continue, dans ce cas, à faire des estimations très précises du nombre de lignes à générer comme montré ci-dessous :

```
SQL> select /*+ gather_plan_statistics *
    / count(1) from t1 where C_DDL = 42;

  COUNT(1)
----------
   3000000
```

```
SQL> select * from table(dbms_xplan.
    display_cursor(null,null,'ALLSTATS
    LAST'));
```

```
-----------------------------------------------------------------------------
| Id  | Operation          | Name | Starts | E-Rows | A-Rows |   A-Time   |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |      1 |        |      1 |00:00:00.37 |
|   1 |  SORT AGGREGATE    |      |      1 |      1 |      1 |00:00:00.37 |
|*  2 |   TABLE ACCESS FULL| T1   |      1 |   2999K|   3000K|00:00:00.44 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(NVL("C_DDL",42)=42)
```

Par contre, en scannant les blocs de la table t1, on remarque l'existence d'un temps d'exécution supplémentaire (44 ms) par rapport à celui de la même opération dans la version d'Oracle précédente (5 ms). Cela est probablement dû à l'application du nouveau filtre qui utilise la fonction NVL :

```
10.2.0.4.0> select /*+
    gather_plan_statistics */ count(1)
    from t1 where C_DDL = 42

  COUNT(1)
----------
   3000000
```

```
10.2.0.4.0> select * from table(
    dbms_xplan.display_cursor(null,null,
    'ALLSTATS LAST'));
```

```
-----------------------------------------------------------------------------
| Id  | Operation          | Name | Starts | E-Rows | A-Rows |   A-Time   |
-----------------------------------------------------------------------------
|   1 |  SORT AGGREGATE    |      |      1 |      1 |      1 |00:00:01.06 |
|*  2 |   TABLE ACCESS FULL| T1   |      1 |   3000K|   3000K|00:00:00.05 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter("C_DDL"=42)
```

3.2 Sur la colonne C_DDL indexée

Lorsqu'une fonction est appliquée sur une colonne figurant dans une clause where de la partie prédicat, elle empêche toute utilisation d'un index qui peut éventuellement exister sur cette colonne. Dans le cas particulier qui nous concerne ici, est-ce que l'utilisation de la fonction NVL appliquée à la colonne va empêcher un index d'être utilisé par le CBO si cette colonne est indexée ? C'est ce que nous allons voir ci-après.

Considérons l'index suivant :

```
SQL> create index i1_c_ddl on t1(c_ddl);

Index created.
Elapsed: 00:00:02.14
```

Et exécutons la même requête encore une fois :

```
SQL> select /*+ gather_plan_statistics *
    / count(1) from t1 where C_DDL = 42;

  COUNT(1)
----------
   3000000
```

```
SQL> select * from table(dbms_xplan.
    display_cursor(null,null,'ALLSTATS
    LAST'));
```

```
SQL> select /*+ gather_plan_statistics */ count(1) from t1 where C_DDL = 42;

  COUNT(1)
----------
   3000000

SQL> select * from table(dbms_xplan.display_cursor(null,null,'ALLSTATS LAST'));

-----------------------------------------------------------------------------
| Id  | Operation          | Name | Starts | E-Rows | A-Rows |   A-Time   |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |      1 |        |      1 |00:00:00.47 |
|   1 |  SORT AGGREGATE    |      |      1 |      1 |      1 |00:00:00.47 |
|*  2 |   INDEX FAST FULL SCAN| I1_C_DDL |      1 |   2999K|   3000K|00:00:00.75 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter("C_DDL"=42)
```

Il y a une bonne nouvelle ici : l'index est utilisé. La fonction cachée NVL n'est pas appliquée sur la colonne C_DDL lorsque celle-ci provient de l'index. Cela explique pourquoi l'index a bien été utilisé par le CBO.

Néanmoins, vous pouvez objecter en disant que c'est un fonctionnement normal et attendu : les valeurs nulles d'une colonne ne sont pas indexées. C'est pourquoi nous allons maintenant créer un index composé de deux colonnes dont l'une est non nulle ; ainsi, toutes les valeurs de la colonne C_DDL, y compris les valeurs nulles, seront indexées. Quelque chose qui ressemble à ce qui suit :

```
SQL> drop index i1_c_ddl;

Index dropped.
```

```
SQL> alter table t1 modify n1 not null;

Table altered.
```

```
SQL> create index i2_n1_c_ddl on t1(n1,
    c_ddl);

Index created.
```

```
SQL> select /*+ gather_plan_statistics *
    / count(1) from t1 where n1= 100 and
     C_DDL= 42;

  COUNT(1)
----------
         1
```

```
-----------------------------------------------------------------------------
| Id  | Operation          | Name | Starts | E-Rows | A-Rows |   A-Time   |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |      1 |        |      1 |00:00:00.01 |
|   1 |  SORT AGGREGATE    |      |      1 |      1 |      1 |00:00:00.01 |
|*  2 |   INDEX RANGE SCAN| I2_N1_C_DDL |      1 |      1 |      1 |00:00:00.01 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("N1"=100 AND "C_DDL"=42)
```

Même lorsque la nouvelle colonne ajoutée C_DDL est protégée contre les valeurs nulles grâce à sa présence dans un index composé, on ne trouve aucune trace de l'utilisation implicite de la fonction NVL appliquée à la colonne C_DDL. Cela démontre clairement qu'à l'inverse des blocs de la table où aucun update de la colonne C_DDL n'a été fait, un index créé sur la même colonne va voir ses blocs feuilles (*leaf blocks*) immédiatement mis à jour par la valeur par défaut de la colonne C_DDL.

Avant de finir ce paragraphe, je vais vous montrer un autre point intéressant à connaître : nous avons vu plus haut qu'à chaque fois que le CBO décide de visiter un bloc de la table, il applique alors la fonction NVL à la colonne C_DDL pour être sûr de rapatrier des valeurs non nulles de cette colonne

(parce qu'en effet, celle-ci n'a pas été mise à jour). Mais nous avons vu que ce filtre est toujours appliqué lorsque la table est totalement scannée (TABLE ACCESS FULL). Est-ce que le CBO va également appliquer cette fonction lorsque la table t1 est accédée via un index (TABLE ACCESS BY INDEX ROWID) ? Je vais donc modéliser un cas très simple pour observer la réaction du CBO dans cette situation particulière :

```
SQL> drop index i2_n1_c_ddl;

SQL> create index i2_n1_c_ddl on t1(n1);

SQL> select /*+ gather_plan_statistics */ count(1) from t1 where n1= 100 and C_DDL= 42;
```

```
-------------------------------------------------------------------------------------
| Id  | Operation                   | Name          | Starts | E-Rows | A-Rows |
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |               |      1 |        |      1 |
|   1 |  SORT AGGREGATE             |               |      1 |      1 |      1 |
|*  2 |   TABLE ACCESS BY INDEX ROWID| T1           |      1 |      1 |      1 |
|*  3 |    INDEX RANGE SCAN         | I2_N1_C_DDL   |      1 |      1 |      1 |
-------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(NVL("C_DDL",42)=42)
   3 - access("N1"=100)
```

Observez comment la fonction NVL a été également appliquée sur la colonne même lorsque la table t1 est visitée via *indexrowid*.

Nous sommes très confiants maintenant de dire qu'à chaque fois que le CBO visite un bloc à partir de la table, que cette visite soit faite via une lecture par bloc unique ou via un accès multibloc, il va appliquer la fonction NVL à la colonne « LDD optimisée » pour filtrer les données prises de ces blocs de table. Par contre, le CBO ne va pas utiliser la fonction NVL sur la colonne C_DDL lorsque celle-ci est acquise à partir d'un bloc feuille d'un index.

4 Oracle 12c et l'optimisation DLL pour les colonnes NULL

Avec l'arrivée de la version Oracle 12c, c'est légitime de se demander si l'optimisation des instructions LDD est encore disponible ou pas. Une image valant mieux que mille mots, essayons aussi la même expérience dans cette nouvelle *release* :

```
12c > alter table t1 add C_DDL number default 42 not null;
Elapsed: 00:00:00.02
```

Presque instantanée, l'optimisation LDD a lieu ici aussi comme montré et prouvé encore une fois par le biais de l'utilisation de la fonction NVL dans la partie prédicat de la requête suivante :

```
12c> select count(1) from t1 where c_ddl=42;

  COUNT(1)
----------
   3000000
```

```
12c> select * from table(dbms_xplan.display_cursor);
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |  3802 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |    13 |            |          |
|*  2 |   TABLE ACCESS FULL| T1   |  3538K|   43M|  3802   (1)| 00:00:01 |
-----------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(NVL("C_DDL",42)=42)

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)
```

Par contre, il y a néanmoins une petite extension de l'optimisation LDD en 12c par rapport à 11gR2. Dans la nouvelle version, cette technique a été étendue pour inclure les colonnes **nulles** ayant une valeur par **défaut**. Considérons la modification de la table suivante faite en 11gR2 et en 12c respectivement pour apprécier clairement la différence :

```
11.2.0.3.0> alter table t1 add C_DDL_2 number default 84;

Table altered.
Elapsed: 00:00:58.25

12c> alter table t1 add C_DDL_2 number default 84;

Elapsed: 00:00:00.02
```

Alors que l'ajout de la colonne C_DDL_2 (qui peut être nulle) a nécessité 58 secondes en 11gR2, il a été accompli instantanément dans la version 12c.

Cela représente une démonstration claire qu'en 12c, l'optimisation des opérations LDD a été étendue aux colonnes **nulles** ayant une valeur par défaut. En effet, lorsqu'on visite la table t1 pour avoir les valeurs distinctes de la nouvelle colonne ajoutée (C_DDL_2), on se rend compte que tous les enregistrements de la table ont vu leurs métadonnées (valeur par défaut 84) mises à jour comme montré par le biais de la requête suivante :

```
12c> select c_ddl_2, count(1) from t1 group by c_ddl_2;

C_DDL_2 COUNT(1)
------- ----------
84       3000000


SQL> select count(1) from t1 where c_ddl_2=84;

  COUNT(1)
----------
   3000000

SQL> select * from table(dbms_xplan.display_cursor);
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |       |       |  3803 (100)|          |
|   1 |  SORT AGGREGATE    |      |     1 |    13 |            |          |
|* 2  |   TABLE ACCESS FULL| T1   |  3538K|    43M|  3803   (1)| 00:00:01 |
-------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(DECODE(TO_CHAR(SYS_OP_VECBIT("SYS_NC00006$",0)),NULL,NVL("
              C_DDL_2",84),'0',NVL("C_DDL_2",84),'1',"C_DDL_2")=84)

Note
-----
   - dynamic statistics used: dynamic sampling (level=2)
```

Cependant, afin d'implémenter l'optimisation LDD pour les colonnes pouvant être nulles, une légère complexité a été introduite par rapport à l'optimisation LDD des colonnes non nulles dans la version précédente d'Oracle. En lieu et place de la simple utilisation de la fonction NVL, est apparu un prédicat exotique et complexe utilisant la fonction d'Oracle SYS_OP_VECBIT non documentée et une colonne interne afin d'honorer la valeur par défaut puisque celle-ci n'a pas été physiquement mise à jour.

Contrairement à ce que vous pouvez immédiatement supposer, la colonne SYS_NC00006$ n'est pas virtuelle. Elle représente une colonne cachée générée intérieurement par Oracle comme montré ci-dessous :

```
12c> SELECT
           column_name
         ,virtual_column
         ,hidden_column
         ,user_generated
     FROM
           user_tab_cols
     WHERE table_name = 'T1'
     AND   column_name = 'SYS_NC0'

COLUMN_NAME          VIR HID USE
-------------------- --- --- ---
SYS_NC00006$         NO  YES NO
```

Bien que cette colonne soit cachée, cela ne nous empêche pas de la sélectionner :

```
12c> select
        a.c_ddl_2
       ,a.SYS_NC00006$
     from t1 a
     where c_ddl_2 =84
           and rownum <=5;

C_DDL_2 SYS_NC00006$
------- ------------
     84
     84
     84
     84
     84
```

La colonne SYS_NC00006$ va rester nulle jusqu'à ce que la colonne C_DDL_2 reçoive une valeur qui n'est pas égale à la valeur par défaut 84. Considérons les inserts suivants :

```
12c> insert into t1 values (0,0,0,'xxxxx
    ',110,130);

1 row created.

12c> insert into t1 values (1,1,1,'xxxxx
    ',140,150);

1 row created.

12c> insert into t1 values (1,1,1,'xxxxx
    ',200,null);

12c> select
        a.c_ddl_2
       ,a.SYS_NC00006$
     from t1 a
     where a.c_ddl_2 in (130,150);

C_DDL_2 SYS_NC00006$
------- ------------
    130 01
    150 01

SQL> select
        a.c_ddl_2
       ,a.SYS_NC00006$
     from t1 a
     where a.c_ddl_2 is null;

C_DDL_2 SYS_NC00006$
------- ------------
        01
```

Observez comment la valeur de la colonne cachée SYS_NC00006$ n'est plus « NULL » lorsqu'on insère une valeur différente de la valeur par défaut 84 (y compris la valeur explicite « NULL »).

En mettant les différentes pièces du puzzle ensemble, nous pouvons très facilement comprendre ce que cet exotique, mais néanmoins simple prédicat, reproduit ci-dessous, est en train de faire exactement :

```
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - filter(DECODE(TO_CHAR(SYS_OP_VECBIT("SYS_NC00006$",0)),NULL,NVL("
              C_DDL_2",84),'0',NVL("C_DDL_2",84),'1',"C_DDL_2")=84)
```

Oracle est simplement en train de vérifier à travers sa colonne système, ainsi que par le biais de l'utilisation de la fonction SYS_OP_VECBIT, s'il doit prendre en considération la valeur par défaut de la colonne ou bien sa vraie valeur introduite par un utilisateur final ou via une requête d'insertion explicite. Essayons d'imiter ce qu'Oracle est en train de faire avec les valeurs 01 et NULL ci-dessus de la colonne SYS_NC00006$:

```
12c> SELECT
          a.c_ddl_2
         ,TO_CHAR(sys_op_vecbit(a.sys_nc0
             0006$,0)) cbo_ddl
     FROM t1 a
     WHERE a.c_ddl_2 IN (130,150)
   UNION ALL
     SELECT
           a.c_ddl_2
          ,TO_CHAR(sys_op_vecbit(a.
              sys_nc00006$,0)) cbo_ddl
     FROM t1 a
     WHERE a.c_ddl_2 IS NULL
   UNION ALL
     SELECT
           a.c_ddl_2
```

```
15          ,TO_CHAR(sys_op_vecbit(a.
                sys_nc00006$,0)) cbo_ddl
16      FROM t1 a
17      WHERE c_ddl_2 =84
18      AND rownum <=1
19      order by c_ddl_2 nulls last
20      ;
21
22  C_DDL_2      CBO_DDL
23  ---------- ---------
24    84         {null}
25    130        1
26    150        1
27   {null}      1
```

Il existe quatre valeurs distinctes de la colonne C_DDL_2, la valeur par défaut (84) et trois autres valeurs insérées explicitement 130, 150 et NULL. Lorsqu'on utilise un prédicat sur la colonne pour rapatrier une ligne à partir d'un bloc de la table, le CBO d'Oracle va décoder la valeur de la colonne CBO_DDL ci-dessus (basée sur SYS_NC00006$) pour vérifier sa valeur par rapport à la valeur du paramètre de la requête (que ce paramètre soit transmis en dur ou en variable de liaison i.e. « literal or bind variable »). Ainsi, Oracle peut imiter correctement toutes les valeurs de la colonne C_DDL_2, y compris celles ayant la valeur par défaut (84) et qui n'ont pas été physiquement mises à jour pour refléter cette valeur par défaut.

5 Conclusion

Oracle 11gR1 a introduit une magnifique nouveauté qui a fait en sorte qu'on ne soit plus jamais inquiet sur la continuité de notre application lorsqu'on projette d'ajouter, en temps réel, une colonne non nulle possédant une valeur par défaut.

Cette nouveauté, appelée « DDL optimization », permet d'ajouter une colonne ayant une valeur par défaut à une table, non seulement instantanément, mais également sans avoir besoin de poser un verrou sur la table altérée. Oracle 12c a étendu cette technique pour y inclure les colonnes nulles avec une valeur par défaut. Et, cerise sur le gâteau, il semble que cela soit sans effets notables sur la performance que l'on est censé observer lors de la sélection de cette colonne ajoutée. En 12c, l'apparition d'un prédicat exotique appliqué lorsque la colonne est lue à partir d'un bloc de la table s'est elle aussi avérée inoffensive et sans effet notable sur la performance d'un select sur les blocs de données de la table.

Mohamed Houri has a PhD in Fluid Mechanics (Scientific Computing) from the University of Aix-Marseille II, preceded by an engineer diploma in Aeronautics. He has been working around the Oracle database for more than 14 years for different Europeans customers as an independent Oracle Consultant specialized in Tuning and Trouble-shooting Oracle performance problems. Mohamed has also worked with the Naval Architect Society of Japan on the analysis of tsunamis and breaking waves using a powerful signal analysis called Wavelet Transform. He maintains an Oracle blog and is active in the Oracle Worldwide forum and in the French equivalent. He tweets about Oracle stuff at @MohamedHouri. Mohamed also is a member of OraWorld Team (www.oraworld-team.com).

Retrouvez l'article de ***Mohamed Houri*** *en ligne : lien 1*

NetBeans

Les dernières news NetBeans

NetBeans 8.0.1 renforce le support de HTML 5, JavaScript et CSS 3 - l'EDI open source disponible

Les développeurs de l'environnement de développement open source NetBeans viennent de publier la version 8.0.1 de l'EDI.

Cette mise à jour pour NetBeans 8 apporte de nouvelles caractéristiques qui concernent essentiellement une meilleure prise en charge du développement Web (HTML 5, JavaScript et CSS 3) dans l'EDI : lien 2.

Au menu des nouveautés, on va noter le support de la modularité et du développement JavaScript orienté entreprise grâce à l'intégration du framework RequireJS. Pour rappel, RequireJS est une implémentation de l'API AMD (Asynchronous Module Definition) permettant d'organiser son code en modules pour optimiser les performances. Toujours côté JavaScript, on va noter la prise en charge du débogage des fichiers JavaScript grâce au projet Karma (solution pour l'exécution des tests). Les modules Node.JS et Bower (outil pour la gestion des dépendances dans des projets JavaScript) peuvent désormais être installés directement à partir de l'EDI.

Les tâches Grunt seront dorénavant disponibles dans le menu contextuel pour les projets Web. La dernière version de PrimeFaces (Framework permettant la création d'interfaces web plus conviviales) a été intégrée dans cette mouture, ainsi que GlassFish 4.1, Tomcat 8.0.9, WildFly et WebLogic 12.1.3.

Les outils de développement pour Java ont été améliorés et l'EDI offre une meilleure prise en charge de Git. À cela, s'ajoutent plusieurs correctifs de bogues.

Cette version sort pratiquement six mois après le lancement de NetBeans 8, dont le numéro de version avait été adopté pour s'aligner avec la sortie de Java 8. NetBeans 8 se présente comme l'outil idéal pour découvrir les nouveautés de Java 8, grâce aux améliorations qui ont été apportées à l'outil et à l'éditeur de code pour travailler convenablement avec les expressions Lambdas, Profiles et Streams.

NetBeans 8 offre également un support complet de Java SE Embedded (création, déploiement, exécution, débogage et profilage des applications Java SE sur des dispositifs embarqués), intègre de nouveaux générateurs de code pour PrimeFaces et offre un meilleur support de JavaScript, HTML5, PHP 5.5 et C++.

La mise à jour NetBeans 8.0.1 est disponible en téléchargement gratuitement.

Télécharger NetBeans 8.0.1 : lien 3

Commentez la news de ***Hinault Romaric*** *en ligne : lien 4*

Java

Les dernières news Java

Java pourrait être doté d'un interpréteur boucle : lecture - évaluation - impression (REPL) - le projet Kulla nait sur OpenJDK

Le JDK sera doté dans une version future de fonctionnalités REPL (Read Evaluate Print Loop). Après un vote qui a eu lieu le 15 septembre dernier, le projet Kulla a été adopté par 14 voix pour, contre 0 vote négatif et aucun veto : lien 5.

La JEP (JDK Enhancement Proposal) du projet est déjà disponible en ligne. Elle servira de feuille de route pour son développement. Elle décrit notamment les objectifs du projet, les motivations, les alternatives, les risques, etc.

Dans ce document, on apprend que la rétroaction immédiate qu'apportera ce projet est importante lors de l'apprentissage d'un langage de programmation. En raison de cette omission, de nombreuses écoles se sont éloignées de Java pour adopter des langages comme Python, qui disposent d'un REPL.

Selon la JEP du projet, ces fonctionnalités feront partie du JDK 9.

Consulter la JEP du projet : lien 6

Commentez la news de ***Hinault Romaric*** *en ligne : lien 7*

Les derniers tutoriels et articles

Tutoriel Ceylon : Typage

Nombreux sont les langages dédiés à la JVM, mais aucun autre que Ceylon ne propose un système de typage aussi poussé ainsi qu'une compilation en JavaScript. À travers une série d'articles, je propose de vous faire découvrir tous les mystères de ce langage conçu par Gavin King : lien 8.

1 Avant-propos

Cet article est le troisième d'une série articulée sur la présentation du langage Ceylon :

- Présentation et installation : lien 9 ;
- Concepts de base : lien 10 ;
- Typage : lien 11 ;
- Appels et arguments ;
- Collections ;
- Modules ;
- Interopérabilité avec Java.

Cet article propose de vous faire découvrir l'un des points forts du langage : son système de typage.

2 Null

Le premier concept à assimiler en Ceylon, c'est que Null (lien 12) est un type à part entière et que null (lien 13)en est simplement le singleton. Cependant Ceylon offre plusieurs sucres syntaxiques pour manipuler les variables qui dépendent de ce type.

2.1 Optionnel

Le premier sucre syntaxique à connaître c'est la notation postfixée ?. Il s'agit simplement d'indiquer qu'une valeur est « optionnelle ». La variable peut être soit du type indiqué, soit du type Null :

```
1  //Source: tutoriel/b_type/a_null/
       a_optionnel.ceylon
2  String? optionnel(Boolean set) {
3    if (set) {
4      return "Une chaîne";
5    } else {
6      return null;
7    }
8  }
9
10 void demoOptionnel() {
11   String? nonNull = optionnel(true);
12   String? null = optionnel(false);
13   print(nonNull);
14   print(null);
15 }
```

Une construction intéressante offerte par Ceylon, c'est le chaînage d'appel. Il arrive parfois que l'on ait besoin de traverser un graphe d'objet, mais que potentiellement, tout ou partie des références à traverser puissent être null. Si l'on veut simplement avoir la valeur null lorsque l'une des références est null, il suffit de l'indiquer à Ceylon :

```
1  //Source: tutoriel/b_type/a_null/
       b_grapheOptionnel.ceylon
2  // Pseudo-graphe dont les éléments
       impairs sont optionnels.
3  class OptionnelA(shared OptionnelB b) {}
4  class OptionnelB(shared OptionnelC? c) {
       }
5  class OptionnelC(shared OptionnelD d) {}
6  class OptionnelD(shared OptionnelE? e) {
       }
7  class OptionnelE(shared String? f) {}
8
9  void afficherOptionnelA(OptionnelA? a) {
10   print(a?.b?.c?.d?.e?.f);
11 }
12
13 void demoGrapheOptionnel() {
14   afficherOptionnelA(null);
15   afficherOptionnelA(OptionnelA(
         OptionnelB(null)));
16   afficherOptionnelA(OptionnelA(
         OptionnelB(OptionnelC(OptionnelD(
         null)))));
17   afficherOptionnelA(OptionnelA(
         OptionnelB(OptionnelC(OptionnelD(
         OptionnelE(null))))));
18   afficherOptionnelA(OptionnelA(
         OptionnelB(OptionnelC(OptionnelD(
         OptionnelE("Résultat"))))));
19 }
```

2.2 Opérateurs

Maintenant que nous pouvons déclarer des références optionnelles, il serait bien de voir comment les manipuler correctement. Le premier couple d'opérateurs est l'un de ceux que l'on a déjà vus lors de l'apprentissage des bases : then-else. else permet soit de récupérer l'opérande de gauche s'il n'est pas null, sinon l'opérande de droite.

```
1  //Source: tutoriel/b_type/a_null/c_else.
       ceylon
```

```
2  void demoNullElse() {
3    String? a = null;
4    String? b = "Une valeur";
5    String? c = "Une autre valeur";
6    print(a else b);
7    print(b else c);
8  }
```

L'opérande gauche de l'opérateur then est une expression logique. Si elle est vraie, l'opérateur retourne l'opérande de droite, sinon la valeur null.

```
1  //Source: tutoriel/b_type/a_null/d_then.
       ceylon
2  void demoNullThen() {
3    String a = "Une valeur";
4    String b = "Une autre valeur";
5    print(true  then a);
6    print(false then b);
7  }
```

Ainsi le but de l'opérateur else est d'empêcher les valeurs null. Tandis qu'au contraire, l'opérateur then produit des valeurs null. Il conviendra donc de les coupler pour éviter les valeurs null :

```
1  //Source: tutoriel/b_type/a_null/
       e_elsethen.ceylon
2  void afficherNullElseThen(Float n) {
3    String signe = (n == infinity then "
         infini")
4    else (n < 0.0 then "négatif")
5    else (n > 0.0 then "positif")
6    else "zéro";
7    print(signe);
8  }
9
10 void demoNullElseThen() {
11   afficherNullElseThen(infinity);
12   afficherNullElseThen(-1.0);
13   afficherNullElseThen(1.0);
14   afficherNullElseThen(0.0);
15 }
```

Un autre opérateur intéressant est exists. Celui-ci renvoie vrai si la référence n'a pas la valeur null. Suite à cela, Ceylon assume que la référence ne peut être null soit dans le bloc pour les structures de contrôles (ex. if, while), soit pour le restant du bloc avec l'instruction assert. On peut ainsi appeler les méthodes d'un type optionnel. L'opérateur peut également être combiné avec une déclaration s'il est utilisé avec une structure de contrôles.

```
1  //Source: tutoriel/b_type/a_null/
       f_exists.ceylon
2  class ClassExists() {
3    shared void demoIf(String? chaine) {
4      if (exists chaine) {
5        print("demoIf : " + chaine.
             uppercased);
6      } else {
7        print("demoIf : NULL");
8      }
9    }
10   shared void demoDeclaration(OptionnelA
         ? a) {
11     if (exists f = a?.b?.c?.d?.e?.f) {
12       print("demoDeclaration : " + f);
13     }
14   }
15   shared void demoAssert(String? chaine)
         {
16     try {
17       assert (exists chaine);
18       print("demoAssert : " + chaine.
             reversed);
19     } catch (AssertionException e) {
20       print("demoAssert : " + e.message)
             ;
21     }
22   }
23 }
24
25 void demoExists() {
26   ClassExists demo = ClassExists();
27   demo.demoIf("fo");
28   demo.demoIf(null);
29   demo.demoDeclaration(OptionnelA(
         OptionnelB(null)));
30   demo.demoDeclaration(OptionnelA(
         OptionnelB(OptionnelC(OptionnelD(
         OptionnelE("ob"))))));
31   demo.demoAssert("ar");
32   demo.demoAssert(null);
33 }
```

3 Type

3.1 Union

L'union de types consiste simplement à indiquer que l'on ne connaît pas le type exact d'une référence, mais que l'on sait qu'il appartient à un ensemble limité. Par exemple, la référence suivante peut être soit une chaîne de caractères, soit un nombre :

```
1  //Source: tutoriel/b_type/a_union/
       a_union.ceylon
2  variable String|Integer variableUnion =
       1;
```

L'union de types est un concept que nous avons déjà manipulé, mais de manière masquée grâce au sucre syntaxique. En effet, la notation postfixée ? est simplement un raccourci pour indiquer l'union du type déclaré et le type Null. Ainsi la déclaration String ? est un raccourci pour String|Null.

De base, Ceylon autorise l'utilisation des méthodes de tous les types communs (classe ancêtre commune, mais également les interfaces. Ainsi dans l'exemple ci-dessous, il est possible d'appeler la méthode afficherA bien que nous ne sachions pas quel

est le type exact de la référence :

```
1  //Source: tutoriel/b_type/a_union/
       b_ancetre.ceylon
2  interface UnionZ {
3    shared default void afficherZ() {
4      print("UnionZ");
5    }
6  }
7
8  class UnionA() {
9    shared void afficherA() {
10     print("UnionA");
11   }
12 }
13
14 class UnionB() extends UnionA()
       satisfies UnionZ {
15   shared void afficher() {
16     print("UnionB");
17   }
18 }
19
20 class UnionC() extends UnionA()
       satisfies UnionZ {
21   shared void afficher() {
22     print("UnionC");
23   }
24 }
25
26 class UnionD() extends UnionA() {
27   shared void afficher() {
28     print("UnionC");
29   }
30 }
31
32 void demoUnion() {
33   UnionB|UnionC union = UnionB();
34   union.afficherA();
35   union.afficherZ();
36 }
```

Pour restreindre une union de types, il faut utiliser l'opérateur is. Comme pour l'opérateur exists, is peut être utilisé dans un if ou un assert et il peut également être accompagné d'une déclaration :

```
1  //Source: tutoriel/b_type/a_union/c_is.
       ceylon
2  void demoIsIf() {
3    if (variableUnion is String) {
4      print("C'est une chaîne");
5    } else if (variableUnion is Integer) {
6      print("C'est un entier");
7    }
8  }
9
10 void demoIsDeclaration() {
11   if (is String variable = variableUnion
         ) {
12     print(variable.uppercased);
13   } else if (is Integer variable =
         variableUnion) {
14     print(variable + 1);
15   }
16 }
17
18 void demoIsAssert() {
19   assert (is Integer variable =
         variableUnion);
20   print(variable - 1);
21 }
```

Il est également possible d'utiliser l'instruction switch avec l'opérateur is :

```
1  //Source: tutoriel/b_type/a_union/
       d_is_switch.ceylon
2  void demoIsSwitch() {
3    String|Integer ref = variableUnion;
4    switch (ref)
5      case (is String) {
6        print(ref.uppercased);
7      }
8      case (is Integer) {
9        print(ref + 1);
10     }
11 }
```

On remarquera dans ce dernier exemple que la référence a dû être « stockée » dans une référence constante avant d'être testée dans le switch afin d'éviter les problèmes de concurrence. Si variableUnion est changé, le type restera bien valide dans les différents blocs du switch.

Pour terminer la présentation des unions de types, je vais vous parler de ce à quoi ils peuvent servir en dehors des références optionnelles. Ceylon n'autorisant pas la surcharge des méthodes et notamment des constructeurs, les unions de types permettent d'autoriser différents types d'arguments. Charge à vous ensuite de distinguer les différents cas :

```
1  //Source: tutoriel/b_type/a_union/
       e_surcharge.ceylon
2  class ClassUnion(String|Integer soi) {
3    shared String s;
4    shared Integer i;
5
6    if (is String ts = soi)       { s = ts
         ; i = 0;  }
7    else if (is Integer ti = soi) { s = ""
         ; i = ti; }
8    else                          { s = ""
         ; i = 0;  }
9
10   shared actual String string => "‘‘s
         ‘‘;‘‘i‘‘";
11
12   void afficherString(String s) {
13     print("afficherString  : " + s);
14   }
15   void afficherInteger(Integer i) {
16     print("afficherInteger : ‘‘i‘‘");
17   }
18   shared void afficher(String|Integer
         soi) {
19     switch (soi)
20       case (is String)  { afficherString
             (soi);  }
21       case (is Integer) {
             afficherInteger(soi); }
22   }
23 }
24
25 void demoClassUnion() {
26   ClassUnion s = ClassUnion("foo");
27   ClassUnion i = ClassUnion(1);
28   print(s);
29   print(i);
```

```
30    s.afficher("bar");
31    s.afficher(2);
32  }
```

3.2 Intersection

Si l'union de types correspond à l'opérateur logique « ou », l'intersection correspond au « et ». En effet, la référence doit correspondre à TOUS les types mentionnés.

```
1  //Source: tutoriel/b_type/b_intersection
       /a_intersection.ceylon
2  interface IntersectionA {
3    shared default void afficherA() {
4      print("IntersectionA");
5    }
6  }
7
8  interface IntersectionB {
9    shared default void afficherB() {
10     print("IntersectionB");
11   }
12 }
13
14 class IntersectionAB() satisfies
       IntersectionA&IntersectionB {}
15
16 void demoIntersection() {
17   IntersectionAB ab = IntersectionAB();
18   ab.afficherA();
19   ab.afficherB();
20 }
```

Ceci permet de gérer des hiérarchies de classes distinctes, mais qui satisfont les mêmes interfaces. L'exemple le plus parlant est le cas des classes StringBuilder (lien 14) et StringBuffer (lien 15) de Java. Leur ancêtre commun est la classe Object (lien 16), cependant elles implémentent toutes deux les interfaces CharSequence (lien 17) et Appendable (lien 18) :

```
1  //Source: tutoriel/b_type/b_intersection
       /DemoAppendableCharSequence.java
2  public class DemoAppendableCharSequence
       {
3
4    public static <T extends Appendable&
         CharSequence> void affiche(T
         appendableCharSequence) throws
         IOException {
5      if (appendableCharSequence.length()
           == 0) {
6        appendableCharSequence.append("<
           empty>");
7      }
8      System.out.println(
           appendableCharSequence);
9    }
10
11   public static void main(String[] args)
       throws IOException {
12     System.out.println(StringBuilder.
         class);
13     affiche(new StringBuilder());
14     System.out.println(StringBuffer.
         class);
15     affiche(new StringBuffer());
16   }
17 }
```

On constate ainsi que l'intersection permet également de contourner la surcharge des méthodes. Plutôt que d'écrire une méthode pour chaque type qui vous intéresse, vous pouvez utiliser une seule méthode qui dépend uniquement d'une liste d'interfaces.

Un point intéressant à noter est l'utilisation de l'opérateur is :

```
1  //Source: tutoriel/b_type/b_intersection
       /b_is.ceylon
2  void demoIntersectionIs() {
3    IntersectionA a = IntersectionAB();
4    a.afficherA();
5    if (is IntersectionB a) {
6      //Ici le type de a est IntersectionA
           &IntersectionB
7      a.afficherA();
8      a.afficherB();
9    }
10 }
```

3.3 Alias

Les alias et l'inférence de type (que nous aborderons par la suite) permettent tous deux de réduire la verbosité. Les alias offrent l'ajout de sémantique au code. Ainsi Gens n'est qu'un ensemble de Personne :

```
1  //Source: tutoriel/b_type/c_alias/a_gens
       .ceylon
2  interface Gens => Set<Personne>;
```

Vous aurez conclu de cet exemple que pour définir un alias, il suffit de préciser le type et le nom suivis d'une « grosse flèche » (fat arrow), et enfin sa définition.

Utiliser un alias n'empêche pas d'utiliser le type désigné :

```
1  //Source: tutoriel/b_type/c_alias/
       b_typeDesigne.ceylon
2  void demoAliasGensAffiche(Gens gens) {
3    print(gens.size);
4  }
5  void demoAliasGens() {
6    Set<Personne> gens = LazySet({Personne
         ("Jean", "DUPONT"), Personne("
         Christine", "DURAND")});
7    demoAliasGensAffiche(gens);
8  }
```

Si vous définissez un alias sur une classe, il faut déclarer les paramètres :

```
1  //Source: tutoriel/b_type/c_alias/
       c_parametre.ceylon
2  class ConstructeurAvecParametre(shared
       String unAttribut) {}
3  class AliasAvecParametre(String
       parametre) =>
       ConstructeurAvecParametre(parametre)
       ;
```

Il est également possible de créer des alias correspondant à des unions, ou des intersections de types à l'aide du mot-clé alias :

```
1  //Source: tutoriel/b_type/c_alias/
       d_union.ceylon
```

```
class A() {}
class B() {}
alias AliasUnionAB => A|B;
```

Il est possible d'étendre ou de satisfaire un alias à condition que celui-ci ne soit pas déclaré en utilisant le mot-clé alias :

```
//Source: tutoriel/b_type/c_alias/e_etendre_satisfaire.ceylon
// Déclaration valide
interface C {
  shared formal String nom;
}
interface D {}
interface AliasC => C;
class AliasClassAvecParametre() extends AliasAvecParametre("a") satisfies C&D {
  shared actual String nom => unAttribut;
}

// Déclaration non valide
alias AliasCetD => C&D;
class AliasClassCetD() satisfies AliasCetD {}
```

Pour finir, un alias peut être générique et avoir des contraintes de types (que nous aborderons plus loin) :

```
//Source: tutoriel/b_type/c_alias/f_generique.ceylon
class Named<Value>(String name, Value val)
        given Value satisfies Object
        => Entry<String,Value>(name,val);
```

3.4 Inférence

L'inférence de type dans Ceylon consiste simplement à omettre le type, d'une référence ou de retour d'une fonction, si :

- celle-ci (la référence ou la fonction) n'est pas exposée (Note : si l'élément est exposé, le langage suppose que cela fait alors partie d'une API) :
  - la déclaration ne peut pas être un élément de haut niveau,
  - la déclaration ne peut pas être partagée (shared),
  - il n'est pas possible d'inférer le type des paramètres ;
- le type peut être déduit dès la déclaration, cela signifie que :
  - la déclaration ne peut pas être abstraite,
  - la référence doit être initialisée lors de sa déclaration.

Si vous remplissez ces conditions, il suffit de remplacer le type par value pour les références et function pour les fonctions.

```
//Source: tutoriel/b_type/d_inference/a_valide.ceylon
class Personne(shared String prenom, shared String nomFamille) {}
void demoInferenceValue() {
  value jeanDupont = Personne("Jean", "DUPONT");
  print(jeanDupont.nomFamille);
}

class DemoInferenceFunction() {
  function somme(Integer a, Integer b) {
    value resultat = a + b;
    return resultat;
  }
  shared void affiche(Integer a, Integer b) {
    print(somme(a,b).sign);
  }
}
```

```
//Source: tutoriel/b_type/d_inference/b_nonvalide.ceylon
// Déclarations de haut niveau
value demoInferenceReferenceDeHautNiveau = "a";
function demoInferenceFonctionDeHautNiveau(Integer a, Integer b) => a+b;

// Déclarations partagées
class DemoInferenceReferencePartagee() {
  shared value reference = "a";
}
class DemoInferenceFonctionPartagee() {
  shared function fonction() {
    return print("a");
  }
}

// Paramètre
class DemoInteferenceParametre() {
  void fonction(value a) {
    print(a);
  }
}

// Déclarations abstraites
abstract class DemoInferenceReferenceAbstraite() {
  formal value reference;
}
abstract class DemoInferenceFonctionAbstraite() {
  formal function fonction();
}

// Initialisation retardée
void demoInferenceInitialisation() {
  value reference;
  reference = "a";
}
```

N'allez pas imaginer que les déclarations ne sont pas typées ! C'est le compilateur qui va s'en charger selon un algorithme très simple. Pour les références, il s'agit du type de l'expression que vous assignez. Pour les accesseurs et les fonctions, il s'agit de l'union des types retournés.

4 Génériques

4.1 Définition

La généricité est la possibilité d'introduire une forme d'abstraction supplémentaire à vos types (classe ou interface). En effet, cela permet de décrire des comportements pouvant exploiter différents types de données. La généricité est la possibilité d'introduire une forme d'abstraction supplémentaire à vos types (classe ou interface). En effet, cela permet de décrire des comportements pouvant exploiter différents types de données.

```
1  //Source: tutoriel/c_generiques/
       a_definition/a_liste.ceylon
2  "Spécification na\"\ive d'une liste"
3  shared interface Liste<Element> {
4    shared formal void add(Element element
         );
5    shared formal Element get(Integer
         index);
6    shared formal Integer size;
7  }
8
9  "Implémentation na\"\ive d'une liste
10  Celle-ci contient une et une seule
       valeur"
11 shared class Singleton<Type>(variable
       Type element) satisfies Liste<Type>
       {
12   shared actual void add(Type newElement
         ) {
13     Boolean unsupported = false;
14     assert (unsupported);
15   }
16
17   shared actual Type get(Integer index)
         {
18     assert(index == 0);
19     return element;
20   }
21
22   shared actual Integer size = 1;
23
24   shared actual String string {
25     if (is Object obj = element) {
26       return "{``obj``}";
27     }
28     return "{}";
29   }
30 }
31
32 void demoGeneriqueListe() {
33   Liste<Integer> unEntier = Singleton<
         Integer>(0);
34   print(unEntier);
35   Liste<String> uneChaine = Singleton<
         String>("une chaîne");
36   print(uneChaine);
37 }
```

La syntaxe suit les conventions d'usage. Une petite particularité par rapport à Java, le type racine n'est pas Object (lien 19), mais Anything (lien 20) ! Toujours pour se distinguer de Java, la convention dans Ceylon est d'utiliser des noms significatifs (ex. « Element ») pour les paramètres de type plutôt qu'une simple lettre (par exemple « E » et List en Java : lien 21).

De la même manière qu'il est possible de définir une valeur par défaut à une fonction, il est également possible de définir un type par défaut à un type générique :

```
1  //Source: tutoriel/c_generiques/
       a_definition/b_defaut.ceylon
2  class GenericDefaut<PremierType,
       SecondType=Null>() {
3    shared void afficherPremier(
         PremierType premier) {
4      String valeur;
5      if (exists premier) { valeur =
           premier.string; }
6      else                { valeur = "";
                           }
7      print("premier=``valeur``");
8    }
9    shared void afficherSecond(SecondType
         second) {
10     String valeur;
11     if (exists second) { valeur = second
           .string; }
12     else               { valeur = "";
                          }
13     print("second=``valeur``");
14   }
15 }
16
17 void demoGenericDefaut() {
18   GenericDefaut<String> chaine =
         GenericDefaut<String>();
19   chaine.afficherPremier("bonjour");
20   chaine.afficherSecond(null);
21 }
```

Les paramètres de type sont obligatoires, ainsi l'écriture suivante n'est pas correcte :

```
1  //Source: tutoriel/c_generiques/
       a_definition/c_obligatoire.ceylon
2  GenericDefaut parametreDeTypeAbsent =
       GenericDefaut<String>();
```

Cependant cette écriture est acceptée si tous les paramètres de type ont une valeur par défaut :

```
1  //Source: tutoriel/c_generiques/
       a_definition/d_facultatif.ceylon
2  class GeneriqueTypeOptionnel<PremierType
       =String>() {}
3  GeneriqueTypeOptionnel
       generiqueTypeOptionnel =
       GeneriqueTypeOptionnel<String>();
```

Si définir les types lors de la déclaration vous ennuie, pensez à utiliser le mot-clé value !

Si les paramètres de type sont obligatoires pour les déclarations de type, ils sont toutefois le plus souvent optionnels lors de l'invocation de méthodes ou d'instanciation :

```
1  //Source: tutoriel/c_generiques/
       a_definition/e_invocation.ceylon
2  class DemoGeneriqueTypeImplicite<Contenu
       >(shared Contenu contenu) {
```

```
3     shared void affiche() {
4       print(contenu);
5     }
6     shared void afficheAutre<AutreContenu>
          (AutreContenu autre) {
7       print(autre);
8     }
9   }
10
11  void demoGeneriqueTypeImplicite() {
12    value generique =
          DemoGeneriqueTypeImplicite("String
          ");
13    generique.affiche();
14    generique.afficheAutre(1);
15  }
```

4.2 Covariance/Contravariance

Passons désormais à un principe plus complexe lié à l'introduction des génériques : covariance/contravariance. Derrière ces mots barbares se cache la possibilité d'utiliser des types génériques avec des paramètres « compatibles ». Conceptuellement, on peut admettre qu'une liste de moutons est une liste d'animaux. En réalité, ce n'est pas strictement vrai, puisqu'on ne pourrait pas ajouter de loup dans cette liste :

```
1   //Source: tutoriel/c_generiques/
        b_variance/a_melange.ceylon
2   import tutoriel.c_generiques.
        a_definition { Liste, Singleton }
3
4   interface Animal {
5     shared default actual String string =>
          "Animal";
6   }
7   interface Mouton satisfies Animal {
8     shared default actual String string =>
          super.string + "/Mouton";
9   }
10  class Dorset() satisfies Mouton {
11    shared default actual String string =>
          super.string + "/Dorset";
12  }
13  class Soay() satisfies Mouton {
14    shared default actual String string =>
          super.string + "/Soay";
15  }
16  class Loup() satisfies Animal {
17    shared default actual String string =>
          super.string + "/Loup";
18  }
19
20  void demoVarianceInvalide() {
21    Liste<Mouton> bergerie      =
          Singleton<Mouton>(Dorset());
22    Liste<Dorset> cheptelDorset = bergerie
          ;  // Et si j'avais une Soay ?
23    Liste<Animal> refuge        = bergerie
          ;  // Supposé bon
24    refuge.add(Loup());
                                  // Un loup
          dans la bergerie !
25  }
```

Bien entendu le compilateur refusera d'affecter bergerie à refuge ou cheptel. On dit alors que Liste est invariante en Element. Nous allons donc séparer les méthodes qui *produisent* de celles qui *consomment* :

```
1   //Source: tutoriel/c_generiques/
        b_variance/b_producteur_consommateur
        .ceylon
2   shared interface Producteur<out Produit>
        {
3     shared formal Produit produire();
4   }
5
6   shared interface Consommateur<in
        Consommable> {
7     shared formal void consommer(
          Consommable p);
8   }
```

— l'annotation out est utilisée pour indiquer que Producteur est covariant en Produit. Ses méthodes ne font que produire (retourner) des Produit mais n'en consomment (prennent en paramètre) jamais ;

— l'annotation in est utilisée pour indiquer que Consommateur est contravariant en Consommable. Ses méthodes ne font que consommer (prendre en paramètre) des Consommable mais n'en produisent (retournent) jamais.

Le compilateur vérifiera si la déclaration in/out des paramètres de type est correcte avec leur utilisation. Ainsi les définitions suivantes ne sont pas valides :

```
1   //Source: tutoriel/c_generiques/
        b_variance/c_non_valide.ceylon
2   interface ProducteurInvalide<out Produit
        > {
3     shared formal void transformer(Produit
          produit);
4   }
5
6   interface ConsommateurInvalid<in
        Consommable> {
7     shared formal Consommable jeter();
8   }
```

Et maintenant ? Puisque le compilateur est sûr de l'utilisation qui est faite des paramètres de type, il nous autorise à considérer :

— un éleveur de moutons comme un éleveur d'animaux. L'espèce qu'il produira sera le mouton et donc un animal. En revanche, il ne pourra pas accepter n'importe quel animal ;

— un loup comme un mangeur de moutons. Il peut manger toute sorte de moutons, mais pas les autres animaux.

```
1   //Source: tutoriel/c_generiques/
        b_variance/d_valide.ceylon
2   void demoVariance() {
3     object eleveurOvin satisfies
          Producteur<Mouton> {
4       shared actual Mouton produire() {
5         return Dorset();
6       }
7     }
8     Producteur<Animal> eleveur =
          eleveurOvin;
```

```
9     print("L'éleveur produit des ``eleveur
          .produire()``");
10
11    object loup satisfies Consommateur<
          Mouton> {
12      shared actual void consommer(Mouton
            mouton) {
13        print("Miam ! Je mange du ``mouton
              ``");
14      }
15    }
16    Consommateur<Dorset> gourmet = loup;
17    gourmet.consommer(Dorset());
18  }
```

4.3 Héritage

En Java, un type ne peut pas hériter d'un autre plus d'une fois, et ce même si les paramètres de type diffèrent :

```
1  //Source: tutoriel/c_generiques/
       c_heritage/a_java.java
2  interface Liste<T> {}
3  class ListeDObject implements Liste<
       Object> {}
4  class ListeDeChaine extends ListeDObject
       implements Liste<String> {}
5  //The interface Liste cannot be
       implemented more than once with
       different arguments: Liste<Object>
       and Liste<String>
```

Tout cela est possible en Ceylon si, et seulement si le type n'est pas invariant en l'un de ses paramètres :

```
1  //Source: tutoriel/c_generiques/
       c_heritage/b_ceylon.ceylon
2  class A() {}
3  class B() {}
4
5  interface HeritageAvecDifferentType<out
       Element> {}
6  interface HeritageAvecDifferentTypeA
       satisfies HeritageAvecDifferentType<
       A> {}
7  interface HeritageAvecDifferentTypeB
       satisfies HeritageAvecDifferentType<
       B> {}
8  class HeritageAvecDifferentTypeAetB()
       satisfies HeritageAvecDifferentTypeA
       & HeritageAvecDifferentTypeB {}
```

Il est intéressant de noter que cette dernière classe est compatible avec HeritageAvecDifferentType<A&B>.

```
1  //Source: tutoriel/c_generiques/
       c_heritage/c_covariance.ceylon
2  void demoHeritageAvecDifferentType() {
3    HeritageAvecDifferentType<A&B> ref =
         HeritageAvecDifferentTypeAetB();
4  }
```

Si les *producteurs* sont covariants en l'intersection des types, les *consommateurs* sont contravariants en l'union des types :

```
1  //Source: tutoriel/c_generiques/
       c_heritage/d_contravariance.ceylon
2  import tutoriel.c_generiques.b_variance
       { Consommateur }
3
4  interface ConsommateurA satisfies
       Consommateur<A> {}
5  interface ConsommateurB satisfies
       Consommateur<B> {}
6  class ConsommateurAetB() satisfies
       ConsommateurA & ConsommateurB {
7    shared actual void consommer(A|B ab) {
8      if (is A a = ab) { print("A=``a``");
           }
9      if (is B b = ab) { print("B=``b``");
           }
10   }
11 }
```

4.4 Union et intersection

Pour aller plus avant avec l'héritage, les unions et intersections, voici quelques propriétés à retenir pour éviter d'être perdu :

- pour les types covariants (« producteurs ») :
  - Producteur<X>|Producteur<Y> est un sous-type de Producteur<X|Y>,
  - Producteur<X>&Producteur<Y> est un super-type de Producteur<X&Y> ;
- pour les types contravariants (« consommateurs ») :
  - Consommateur<X>|Consommateur<Y> est un sous-type de Consommateur<X&Y>,
  - Consommateur<X>&Consommateur<Y> est un super-type de Consommateur<X|Y>.

4.5 Contraintes

Bien que nous ayons la possibilité de définir des types et des fonctions génériques, il est souhaitable d'avoir un minimum de suppositions concernant les types manipulés. Par exemple, un ensemble n'a de sens que si l'égalité est définie. Or, celle-ci est définie à partir du type « Object » : lien 22.

```
1  //Source: tutoriel/c_generiques/
       e_contraintes/a_type.ceylon
2  class Ensemble<out Element>(Element
       element) given Element satisfies
       Object {
3    shared Boolean contient(Object test) {
4      return element == test;
5    }
6  }
```

5 Énumération

Les types d'un langage comme Ceylon peuvent être vus comme des ensembles dont les éléments sont les instances. Les énumérations de Ceylon permettent de définir des ensembles disjoints.

Les énumérations sont toutes introduites à l'aide du mot-clé of suivi de l'ensemble des sous-types séparés par un « *pipe* » (|) :

```
1  //Source: tutoriel/d_enumeration/
       a_declaration.ceylon
2  interface A4Roues {}
3  class Vehicule() {}
4  abstract class VehiculeA4Roues() of
       Voiture|Camion extends Vehicule()
       satisfies A4Roues {}
5  class Voiture() extends VehiculeA4Roues
       () {}
6  abstract class Camion() extends
       VehiculeA4Roues() {}
7  class Semi() extends Camion() {}
```

Voici quelques règles au sujet des énumérations :

1. Le type énumérant (i.e. VehiculeA4Roues) doit être abstrait (abstract class ou interface) afin de s'assurer que les instances appartiennent nécessairement à l'une des classes filles (ensembles disjoints) ;
2. Le type énumérant peut implémenter des interfaces (i.e. A4Roues) et/ou hériter d'une classe abstraite ou non (i.e. Vehicule) ;
3. Les types énumérés (i.e. Voiture|Camion) doivent tous exister dans le même module (mais pas nécessairement le même package) ;
4. Les types énumérés doivent explicitement étendre ou satisfaire le type énumérant ;
5. Un type énuméré peut être abstrait et avoir un nombre indéfini de types enfants.

Les énumérations sont nécessaires aux conditions d'un switch. En effet, dans ce dernier tous les cas doivent être disjoints, le débranchement ne doit pas tenir compte de l'ordre des cas (contrairement à une succession de if). Sur une hiérarchie de classes, il n'y a aucun souci, car le système de type connaît la relation entre deux classes :

```
1  //Source: tutoriel/d_enumeration/
       b_switch_classes.ceylon
2  class Racine() {}
3  class Enfant1() {}
4  class Enfant2() {}
5  class Feuille1() extends Racine() {}
6  class Feuille2() extends Enfant1() {}
7  class Feuille3() extends Enfant1() {}
8  class Feuille4() extends Enfant2() {}
9  class PetitEnfant() extends Enfant2() {}
10 class Feuille5() extends PetitEnfant() {
       }
11 class Feuille6() extends Enfant2() {}
12
13 void switch_sur_classes() {
14   {Object+} liste = {
15       Racine(), Enfant1(), Enfant2(),
           Feuille1(), Feuille2(), Feuille
           3(),
16       Feuille4(), PetitEnfant(), Feuille5
           (), Feuille6()
17   };
18   for (Object objet in liste) {
19     String type;
20     switch (objet)
21       // case (is Racine) {}
22       case (is Enfant1)  { type = "le
             premier enfant";    }
23       // case (is Enfant2) {}
24       case (is Feuille1) { type = "la
             première feuille";  }
25       // case (is Feuille2) {}
26       // case (is Feuille3) {}
27       case (is Feuille4) { type = "la
             sixième feuille";   }
28       // case (is PetitEnfant) {}
29       case (is Feuille5) { type = "la
             cinquième feuille"; }
30       case (is Feuille6) { type = "la
             sixième feuille";   }
31       else               { type = "un
             inconnu";           }
32     print("Je suis ``type``");
33   }
34 }
```

En revanche, pour les interfaces ce n'est pas possible, car elles ne forment pas normalement de hiérarchie. L'utilisation de l'énumération devient alors obligatoire :

```
1  //Source: tutoriel/d_enumeration/
       c_switch_interfaces.ceylon
2  interface Noeud of Fichier | Repertoire
       | Lien {
3    shared formal String nom;
4  }
5  interface Fichier    satisfies Noeud {}
6  interface Repertoire satisfies Noeud {
7    shared formal {Noeud*} noeuds;
8  }
9  interface Lien       satisfies Noeud {
10   shared formal Noeud cible;
11 }
12
13 class FichierImpl   (shared actual
       String nom
                                  )
       satisfies Fichier {}
14 class RepertoireImpl(shared actual
       String nom, shared actual Noeud[]
       noeuds) satisfies Repertoire {}
15 class LienImpl      (shared actual
       String nom, shared actual Noeud
       cible   ) satisfies Lien {}
16
17 "
18   racine
19     noeud_1/
20       noeud_1_1
21       noeud_1_2
22       noeud_1_3/
23     noeud_2/
24       noeud_2_1 -> noeud_1
25 "
26 void switch_sur_interfaces() {
```

```
27    Noeud noeud_1_1 = FichierImpl("noeud_1
          _1");
28    Noeud noeud_1_2 = FichierImpl("noeud_1
          _2");
29    Noeud noeud_1_3 = RepertoireImpl("
          noeud_1_3", []);
30    Noeud noeud_1   = RepertoireImpl("
          noeud_1", [noeud_1_1, noeud_1_2,
          noeud_1_3]);
31    Noeud noeud_2_1 = LienImpl("noeud_2_1"
          , noeud_1);
32    Noeud noeud_2   = RepertoireImpl("
          noeud_2", [noeud_2_1]);
33    Noeud racine    = RepertoireImpl("
          racine", [noeud_1, noeud_2]);
34    parcourir("", racine);
35  }
36
37  void parcourir(String prefixe, Noeud
        courant) {
38    switch (courant)
39      case (is Fichier) {
40        print(prefixe + courant.nom);
41      }
42      case (is Repertoire) {
43        print(prefixe + courant.nom + "/")
              ;
44        String sousPrefixe = prefixe + "
              ";
45        for (Noeud noeud in courant.noeuds
              ) {
46          parcourir(sousPrefixe, noeud);
47        }
48      }
49      case (is Lien) {
50        print(prefixe + courant.nom + " ->
               " + courant.cible.nom);
51      }
52      else {}
53  }
```

Bien entendu, il ne sera pas possible de satisfaire deux interfaces énumérées :

```
1  //Source: tutoriel/d_enumeration/
       d_satisfaires_enumerees.ceylon
2  // Erreurs de compilation
3  class FichierRepertoire1(shared actual
       String nom, shared actual Noeud[]
       noeuds) satisfies Fichier&Repertoire
        {}
4  class FichierRepertoire2(String nom,
       shared actual Noeud[] noeuds)
       extends FichierImpl(nom) satisfies
       Repertoire {}
```

Désormais on est en droit de se demander comment écrire une énumération de valeurs comme en Java ? Eh bien, en faisant une énumération d'objets :

```
1  //Source: tutoriel/d_enumeration/
       e_enum_objets.ceylon
2  class Forme() {}
3  abstract class FormeSimple() of carre |
       rond | triangle extends Forme() {}
4  object carre    extends FormeSimple() {}
5  object rond     extends FormeSimple() {}
6  object triangle extends FormeSimple() {}
```

Cerise sur le gâteau, contrairement à Java, vos énumérations Ceylon peuvent étendre une classe !

Lorsque vous utilisez des objets dans un switch, la syntaxe est un peu différente. En effet le mot-clé is est omis :

```
1  //Source: tutoriel/d_enumeration/
       f_switch_objets.ceylon
2  abstract class FormeComplexe() of Chemin
       | Polygone extends Forme() {}
3  class Chemin()   extends FormeComplexe()
       {}
4  class Polygone() extends FormeComplexe()
       {}
5
6  void switchObjets() {
7    Forme forme = carre;
8    switch (forme)
9      case (carre, triangle)    { print("J
           'ai des angles");
                                    }
10     case (rond)               { print("
           Je n'ai pas d'angles");
                                   }
11     case (is Chemin|Polygone) { print("J
           'ai trop de complexes");
                                 }
12     else                      { print("
           Angle ou pas angle, telle est la
            question"); }
13 }
```

Avant d'en terminer avec les énumérations, je souhaitais indiquer qu'en plus de fiabiliser votre code, elles permettent de protéger vos API en empêchant les utilisateurs de créer de nouvelles sous-classes.

6 Conclusion

Désormais vous devriez être en mesure d'exploiter au mieux les capacités du typage de Ceylon. La prochaine partie est consacrée à la gestion des appels et des arguments.

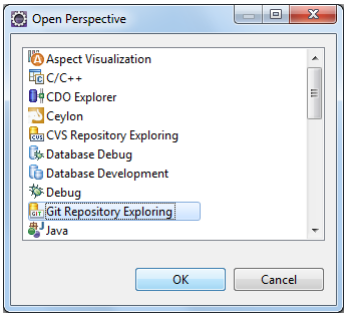
7 Annexes

7.1 Sources des exemples

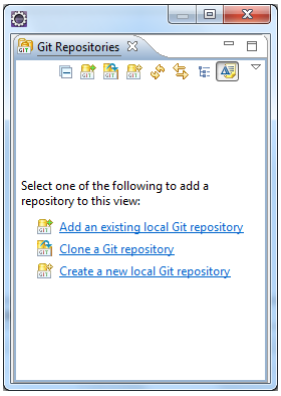
Tous les exemples donnés dans cet article sont disponibles sous GitHub dans le répertoire src-03-typage : lien 23.

7.2 Importer le projet sous Eclipse depuis Git

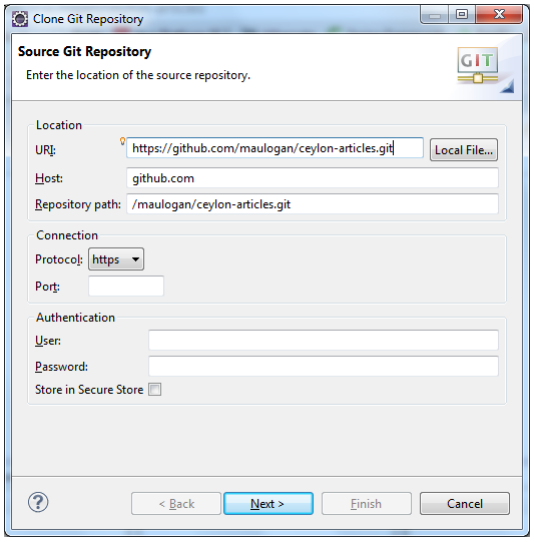
Pour importer le projet sous Eclipse, ouvrez la perspective « Git Repository Exploring » :



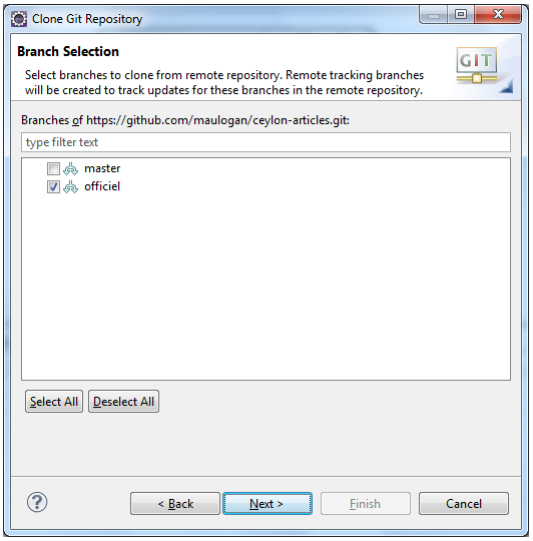
Copiez l'URL https://github.com/maulogan/ceylon-articles.git, assurez-vous que la vue active est « Git Repositories » (en cliquant sur l'onglet, par exemple) :



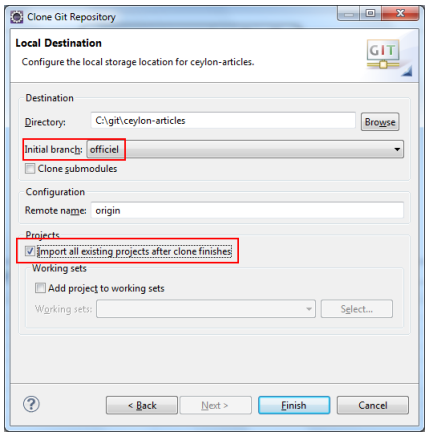
Appuyez simplement sur la combinaison de touche « CTRL+V » pour faire apparaître la fenêtre « Clone Git Repository » :



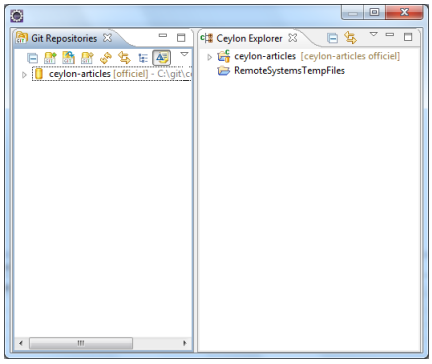
Après avoir appuyé sur « Next > », la fenêtre de sélection des branches apparaît. Sélectionnez uniquement « officiel » :



Appuyez sur « Next > » pour faire apparaître la fenêtre de configuration du stockage local. Adaptez le chemin selon vos préférences, puis vérifiez que la branche est bien « officiel » et activez l'import des projets existants :



Il n'y a plus qu'à terminer en cliquant sur « Finish ». Le dépôt est cloné, puis le projet « ceylon-articles » est importé dans l'espace de travail :



Retrouvez l'article de ***Logan Mauzaize*** *en ligne : lien 24*

Android

Les derniers tutoriels et articles

À la découverte du SDK de compression de bibliothèque native pour les applications Android

1 Introduction

Les applications Android sont généralement écrites en Java en raison de sa conception orientée objet élégante, et du fait que le kit de développement logiciel Android (Android SDK) fournit des fonctionnalités multiplates-formes en Java. Cependant, les développeurs doivent parfois utiliser l'interface native d'Android, surtout si la gestion de la mémoire et les performances sont critiques. L'interface native d'Android est fournie par le kit de développement natif (NDK), permettant le développement natif en C/C++.

Il existe de nombreuses applications NDK sur le magasin de logiciels de Google. Pour réduire la taille du paquet, certains développeurs de logiciels fournissent un seul APK, mais pas un APK FAT (un seul APK contient soit un ARM, soit une bibliothèque x86 tandis qu'un APK FAT intègre les deux à la fois). En effet, les APKs FAT ont des avantages par rapport aux APKs simples. Ils sont plus accessibles pour les utilisateurs finaux et les bibliothèques ne se chevauchent pas lors de la mise à jour de l'application. Les développeurs sont donc encouragés à choisir des APKs FAT pour l'écosystème de logiciels x86 Android. Mais comment les développeurs peuvent-ils réduire la grande taille des fichiers APK FAT ?

2 Zip* vs. LZMA

Pour résoudre le problème de la taille de l'APK FAT, l'auteur a développé un SDK de compression de bibliothèque native. L'idée est d'utiliser l'algorithme de chaîne Lempel-Ziv-Markov (LZMA) (lien 25) pour compresser les bibliothèques natives. Google utilise Zip pour compresser tout le contenu, et malgré que Zip soit rapide, son taux de compression est inférieur à LZMA. D'ailleurs, LZMA est particulièrement efficace pour la compression des volumineux fichiers de dictionnaires qui se trouvent dans les bibliothèques natives. Le graphique suivant montre la différence dans la taille des fichiers après avoir effectué des compressions avec Zip et LZMA.

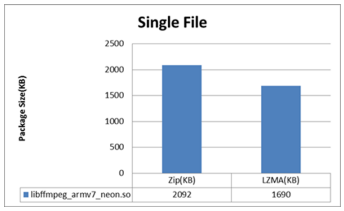


Figure 1 : comparaison de la taille d'un fichier compressé avec Zip * et LZMA.

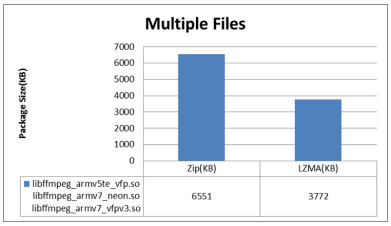


Figure 2 : comparaison de la taille de plusieurs fichiers (même architecture de CPU) compressés avec Zip * et LZMA.

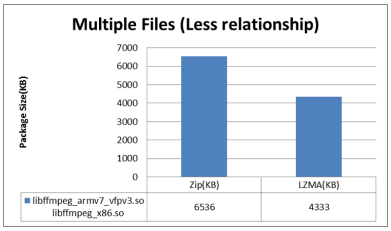


Figure 3 : comparaison de la taille de plusieurs fichiers (architecture CPU différente) compressés avec Zip * et LZMA.

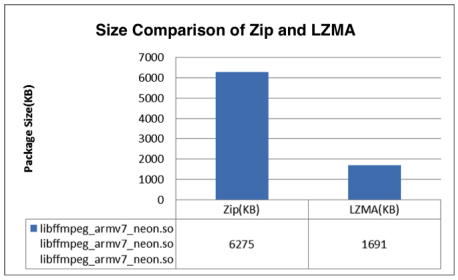


Figure 4 : comparaison de la taille de trois fichiers identiques après avoir été compressés avec Zip et LZMA.

À partir des quatre graphiques ci-dessus, nous pouvons conclure que LZMA peut réduire la redondance entre les fichiers. Dans les cas extrêmes (trois fichiers identiques), LZMA peut obtenir un taux de compression plus élevé que Zip. Cette fonctionnalité est particulièrement adaptée pour la compression de bibliothèques natives. Généralement, une bibliothèque native utilise le même code pour obtenir « armeabi », « armeabi-V7A », « x86 » ou même les bibliothèques « mips ». Pour « armeabi-V7A », il y a un code ARM NEON et un code non-NEON. Ces bibliothèques ont des redondances dues au même code source. Même avec une architecture différente, par exemple libffmpeg_armv7_vfpv3.so et libffmpeg_x86.so, quand l'un est ARMEABI et l'autre x86, le taux de compression est de 40 % tandis que pour un seul fichier, le taux n'est que de 20 %.

3 SDK de compression de bibliothèque native

Le SDK de compression de bibliothèque native développé par l'auteur peut aider les développeurs d'applications dans l'intégration de la compression LZMA de bibliothèque native, et ce pour obtenir des fichiers plus petits. L'idée de base de ce SDK est de compresser l'ensemble de la bibliothèque native dans le dossier « asserts ».

L'API dans ce SDK est très simple. À la première exécution de cette application, le code extrait la totalité de la bibliothèque native du dossier « asserts ».

```
1 static boolean NewInstance(Context cont,
      Handler hdl,boolean showProgress)
2 static DecRawso GetInstance()
3 String GetPath(String libname)
```

Il est recommandé que les développeurs d'applications modifient le code source et ajoutent seulement NewInstance au démarrage de l'application, puis changent system.loadLibrary(***) en System.load (DecRawso. GetInstance(). GetPath(***)). Après ces changements mineurs, l'APK sera plus petit et s'exécutera comme précédemment.

Si les développeurs peuvent retarder suffisamment l'appel de NewInstance jusqu'au premier chargement de bibliothèque native, ils doivent appeler (NewInstance(cont, null, false)) sans le Handler. Sinon, un Handler doit être ajouté pour recevoir le message asynchrone "decode end".

L'auteur a intégré ce SDK dans MoboPlayer sur une tablette de processeur Intel ® Atom ™ Z2760 (nom de code Clover Trail). Le code appelle NewInstance dans la méthode de synchronisation lorsque les utilisateurs démarrent l'application et l'écran de démarrage est affiché. L'appel de NewInsance est transparent pour l'utilisateur final, car le décodage est effectué en arrière-plan. Le tableau suivant montre le résultat de la compression :

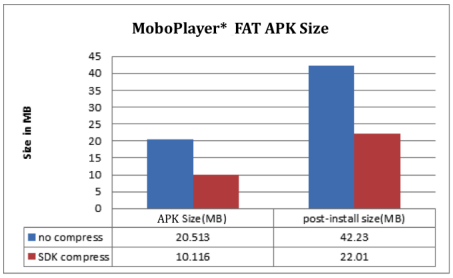


Table 5 : taille de la compression du FAT APK MoboPlayer.

4 Fonctions améliorées du SDK de compression de bibliothèque native

Outre la compression LZMA, ce SDK fournit des fonctions supplémentaires pour encourager les développeurs à inclure une bibliothèque native x86 comme suit :

1. **Le stockage sur Cloud :** les éditeurs de logiciels peuvent stocker les bibliothèques x86 sur un serveur et ces dernières peuvent être téléchargées à partir du serveur après l'installation. Cette action se fait automatiquement sur des appareils x86 et n'est activée que lorsque le Wi-Fi est connecté ;
2. **Détection des bibliothèques manquantes :** si les bibliothèques x86 sont manquantes, le SDK va automatiquement réextraire les bibliothèques ARM. Un développeur peut copier la bibliothèque ARM dans le dossier x86 pour éviter ce problème, mais il faut s'assurer qu'il n'y ait pas de référence croisée entre les bibliothèques ARM et x86 ;
3. **L'outil de construction de paquets Java :** un outil de construction de paquet Java est fourni pour convertir les APKs normaux

en APKs de LZMA compressés. Cet outil prend en charge les systèmes Windows, Linux, Mac. Vous pouvez l'utiliser comme suit : ComPressApk.jar-C :/ mon test.APK-kc :/ clé / *** # # # alias, si "-k" est absent (c'est-à-dire, vous pouvez simplement appeler ComPressApk . pot-C :/ mon / test.APK), la touche de test par défaut « Eclipse » sera utilisée pour signer cet APK. Cet outil peut être intégré dans un script de construction « ants » pour gérer automatiquement la compilation et la publication ;

4. **Le filtre :** la fonction ConfigureFilter vous permet seulement d'extraire les bibliothèques nécessaires. Par exemple, enteringConfigureFilter("libffmpeg", "libffmpeg_armv7_neon.so") signifie qu'il faut extraire uniquement libffmpeg_armv7_neon.so parmi toutes les bibliothèques qui commencent par « libffmpeg ». Ceci est utile pour réduire la taille de la post-installation.

5 Conclusion

L'utilisation du SDK de compression de bibliothèque native pour vos applications Android peut réduire considérablement la taille du paquet de la bibliothèque native et permettre aux applications d'utiliser de grandes bibliothèques natives (en général, ces applications sont des lecteurs vidéo, navigateurs et des jeux en 3D). Pour le code source et le support technique, merci de contacter l'auteur.

Toutes les infos et outils sur Intel Developer Zone Android : lien 26.

Retrouvez l'article de ***Yuming Li*** *traduit par* ***Lana Bauer*** *en ligne : lien 27*

2D/3D/Jeux

Les derniers tutoriels et articles

OpenGL Moderne - Compteur de FPS

Dans le monde des graphismes en temps réel, il est important de garder un œil sur les performances. Une bonne pratique est de déterminer un nombre de FPS comme objectif (habituellement 60 ou 30) et faire tout son possible pour s'y tenir.

1 Un compteur de FPS

Dans le monde des graphismes en temps réel, il est important de garder un œil sur les performances. Une bonne pratique est de déterminer un nombre de FPS comme objectif (habituellement 60 ou 30) et faire tout son possible pour s'y tenir.

Un compteur de FPS ressemble à cela :

```
1  double lastTime = glfwGetTime();
2   int nbFrames = 0;
3
4   do{
5
6       // Mesure la vitesse
7       double currentTime = glfwGetTime();
8       nbFrames++;
9       if ( currentTime - lastTime >= 1.0
            ){ // Si le dernier prinf() a é
            té réalisé il y a plus d'une
            seconde
10          // printf et réinitialise le
                chronomètre
11          printf("%f ms/frame\n", 1000.0/
                double(nbFrames));
12          nbFrames = 0;
13          lastTime += 1.0;
14      }
15
16      ... la suite de la boucle
            principale
```

Il y a une chose étrange dans ce code. Il affiche le temps, en millisecondes, nécessaire pour afficher une image (la moyenne sur une seconde) au lieu du nombre d'images dessinées durant la dernière seconde.

En fait, c'est **beaucoup mieux**. Ne faites **pas** confiance aux FPS. FramePerSecond = 1/SecondPerFrame, ce qui est une proportionnelle inverse et les humains sont nuls pour comprendre ce genre de proportionnelle. Prenez un exemple : vous écrivez une superbe fonction de rendu qui fonctionne à 1000 FPS (1ms/image), mais vous oubliez un petit calcul dans le shader, qui ajoute un coût de 0,1 ms. Et bam, 1/0.0011 = 900, vous venez de perdre 100 FPS. Moralité : **n'utilisez jamais les FPS pour des analyses de performances**.

Si vous prévoyez de faire un jeu fonctionnant à 60 FPS, votre objectif sera 16,6666 ms ; si vous prévoyez de faire un jeu fonctionnant à 30 FPS, votre objectif sera 33,3333 ms. C'est tout ce que vous devez savoir.

Ce code est disponible dans tous les tutoriels à partir du neuvième : indexation VBO (lien 28) ; regardez le fichier *tutorial09_vbo_indexing/tutorial09.cpp*. D'autres outils pour les performances sont disponibles dans la page Outils - débogueurs : lien 29.

*Retrouvez l'article d'**Opengl-tutorial.org** traduit par **Alexandre Laurent** en ligne : lien 30*

Week-end de création de jeux vidéo du 22 au 24 août 2014 - Compte rendu des participations

Du 22 au 24 août 2014 s'est tenu le quatrième week-end de création de jeux vidéo sur le chat de Developpez.com. Comme pour les deux éditions précédentes (que vous pouvez retrouver ici (lien 31) et là (lien 32)), ce fut l'occasion pour les participants de créer un jeu dans un laps de temps très limité.

1 Introduction

Un week-end, c'est seulement 48 heures pour faire un jeu vidéo. Voici le défi qu'ont relevé une dizaine de membres du forum !

Le principe est très simple. Il suffisait de programmer. Le choix de la technologie était libre, tout comme le choix du jeu. Ainsi, nous pouvons énumérer différentes technologies et langages pour la réalisation des jeux, tels que : C++, Java, Flash, HTML5... L'événement avait pour lieu de rencontre le chat de Developpez.com (lien 33), où les participants racontaient l'avancement de leur projet ou demandaient de l'aide. Il n'y avait donc aucune contrainte, le but principal était de s'amuser.

Durant le week-end, nous avons vu plus de trente personnes dans le salon spécialement créé pour l'événement (ce qui est incroyable pour un week-end).

Retrouvez cet événement sur le forum : lien 34.

2 Participations

Vous pouvez télécharger le pack des participations (certains jeux ne sont consultables que sur Internet et donc non présents dans le pack) : lien 35.

2.1 Bloc Gnop par Fusoy

Page du projet : lien 36

Description :

Vous contrôlez une raquette, des blocs tombent irrémédiablement et vous devez empêcher ces blocs d'atteindre le bas en les renvoyant avec votre raquette. Dans ce jeu de scoring, deux modes sont disponibles : 60 secondes et 5 vies. Arriverez-vous à faire sauter les scores ?

Téléchargement : Windows : lien 37. Linux : lien 38.

Retour sur l'événement :

C'était vraiment cool. Voir tout le monde faisant son jeu avec des technologies et approches différentes, c'est très enrichissant. Je participerai volontiers à une prochaine édition.

2.2 tkBoulderDash par Raphaël Seban (tarball69)

Page du projet : lien 39

Description :

Variante (un peu) improvisée du célèbre jeu Boulder Dash des années 80, une création de First Star Software Corp.

Un mineur doit collecter des diamants dans une mine, mais plus il creuse et plus les rochers risquent de lui tomber sur la tête !

On passe au niveau suivant à chaque fois qu'on a collecté tous les diamants présents dans le niveau. Un compteur de diamants restants permet de savoir où on en est.

Petit truc : si on clique dans l'écran de jeu et qu'on déplace la flèche de la souris en maintenant le bouton enfoncé, on peut visiter virtuellement la mine pour voir où se cachent les derniers diamants...

Téléchargement : lien 40.

Retour sur l'événement :

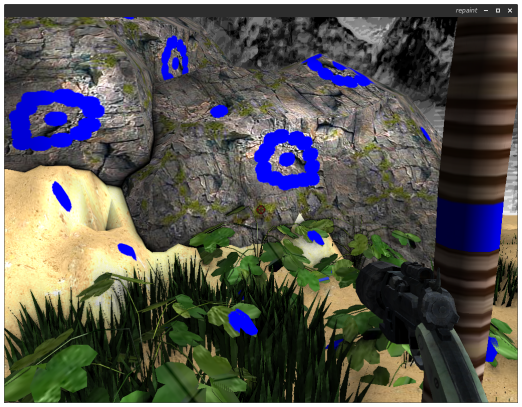
Super sympa. À refaire : une fois vers les fêtes de fin d'année, une fois en été.

2.3 Repaint par Johann Sorel (eclesia)

Page du projet : lien 41

Description :

Ce jeu est un dérivé de FPS. Le but n'étant pas de tuer des ennemis, mais de colorier une image a l'aide d'un paintball. Le jeu n'étant pas fini, il s'agit davantage d'une démo technique du moteur 3D de la bibliothèque Unlicense-Lib (lien 42).



Téléchargement : Jeu : lien 43. Source : lien 44.
Retour sur l'événement :
Sympathique, mais aucun autre développeur sur de la 3D, dommage.

2.4 Tetrisnet par forthx

Page du projet : lien 45
Retour sur l'événement :
Pas de version jouable pour moi cette année, j'en retire juste une belle expérience de programmation réseau. Je suis un peu déçu d'avoir dû jeter l'éponge, mais j'admets que je ne me suis pas réellement donné les moyens : comme les années précédentes, je suis parti de peu pour ne pas dire de rien. Je n'ai pu libérer que 10 h et compte tenu de mes ambitions, cela n'a pas suffi. Malgré tout, j'ai apprécié l'événement et je vois ça comme un tremplin pour une prochaine réalisation.

2.5 The Adventures of Link par Light99

Page du projet : lien 46
Description :
Le but du jeu est de capturer tous les rubis se trouvant sur la carte (en évitant le méchant très très rapide) et de finir la partie en allant sur le marquage orange.
Téléchargement : Version hors ligne : lien 47. Version en ligne : lien 48.
Retour sur l'événement :
J'ai vraiment apprécié cet événement, je tiens à y participer l'année prochaine.

2.6 Cara Cara Carambar par Guntha

Page du projet : lien 49
Description :
Faufilez une barre tournoyante au milieu de la circulation routière sans l'écraser contre les voitures qui vous entourent. Concept repris principalement de Kuru Kuru Kururin.
Téléchargement : Version Windows : lien 50. Version Ouya : lien 51.
Retour sur l'événement :
Cette année, on a eu un excellent week-end de développement, avec beaucoup de productions de qualité, certaines même de qualité professionnelle, et plein de gens sympas présents sur le chat. Me concernant, le projet que j'ai fait cette année est celui des trois auxquels j'ai participé qui me rend le plus fier (j'ai participé aux deux premiers week-ends, pas au troisième). D'un autre côté, cette année je ne suis pas vraiment parti de zéro (j'avais un début de moteur à ma disposition et quelques graphismes que j'ai réutilisés et/ou adaptés, les deux autres fois j'étais parti sur du Java avec Java2D, et j'avais dessiné tout ce dont j'avais besoin pour l'occasion). C'est enrichissant de voir qu'avec somme toute assez peu de participants, on assiste à l'utilisation de beaucoup de technologies différentes. J'espère que je serai disponible aux dates du prochain week-end.

2.7 Opus Magnum par Sylvain Pollet-Villard

Page du projet : lien 52
Description :
Maître des quatre éléments, la Terre, le Feu, l'Air et l'Eau, vous partez explorer la Forêt Sacrée pour récupérer les Grands Éléments et réaliser l'Opus Magnum.
Téléchargement : Version en ligne : lien 53.
Retour sur l'événement :
Des participants plus nombreux et plus présents que l'année précédente, il y a eu du monde sur le chat toute la nuit ! Certains utilisaient Twitter et Twitch, ça a beaucoup aidé à donner de la visibilité à l'événement. Malgré la différence de technos, niveau et styles de jeu, l'ambiance fut très bonne. J'espère qu'elle sera encore meilleure au prochain WEJV5 !

2.8 WCOM par Wahid Garci (ShadowTzu)

Page du projet : lien 54
Description :
Ceci est ma première participation au Ludum Dare. J'ai utilisé VB.NET et ma propre bibliothèque graphique : Tzu3D (VB.NET + SlimDX (DX11)) pour réaliser mon jeu.

Votre but est de trouver les artefacts, qui une fois assemblés, permettront la construction pour déformer le temps et donner l'accès au voyage interstellaire.

Votre équipement est constitué de :

— un pistolet ;
— une mitrailleuse.

Il doit être pris en compte que dans l'urgence, il est impossible de se soigner entre chaque mission. Bonne chance soldat, l'humanité tout entière compte sur vous ! Nécessite :

— Windows 7 (64 bits)
— .NET 4.5

Voir la vidéo : lien 55.
Téléchargement : Jeu : lien 56.

2.9 Caitland par Meiryo

Page du projet : lien 57
Description :
Aventure en lignes de commande. Vous vous réveillez allongé près d'un lac, et allez devoir explorer les environs pour découvrir ce qui a bien pu vous arriver... Full HTML+Java Script
Téléchargement : Version en ligne : lien 58. Source : lien 59.
Retour sur l'événement :
Excellent ! Bon évidemment j'ai fini avec un jour complet de retard, donc c'est pas brillant à ce niveau-là, mais c'est très intéressant de se prouver qu'on est capable de faire quelque chose de pas trop mal en si peu de temps. Je referai sans doute le prochain, en ayant mieux les idées en place avant de commencer le week-end de coding.

2.10 NULLOTRON par CodeurPlusPlus

Page du projet : lien 60
Description :
Nullotron est un jeu dans lequel vous devez débarrasser l'écran d'un grand nombre de monstres en même temps. On choisit la direction de déplacement à l'aide des touches fléchées et la direction de tir à l'aide des touches d, x, c et v (haut, gauche, bas et droite respectivement). NULLOTRON contient la version exécutable sous Windows que j'ai rendue dimanche soir. Cette version a quelques bogues dont un vraiment gênant : certains ennemis disparaissent tout seuls !

HARDNULLOTRON contient deux versions du jeu :

— NTRON.exe est la version de dimanche avec le bogue ci-dessus corrigé (le jeu est beaucoup mieux ainsi).

— HARDNTRON.exe est une version très légèrement modifiée qui est beaucoup plus frénétique (la version de base s'agite déjà de partout).

Téléchargement : NULLOTRON : lien 61. HARDNULLOTRON : lien 62.
Retour sur l'événement :
Je n'avais pas participé les années précédentes, mais si l'ambiance était aussi bonne, j'ai raté quelque chose ! Cette année c'était génial : défi intéressant, grande liberté de choix du jeu qu'on décide de programmer, discussions variées, drôles et parfois pointues techniquement sur le chat... et puis ça m'a donné envie de programmer d'autres jeux plus ambitieux ! Je reviendrai dans un an pour la cinquième édition.

2.11 Super Mandala Omega par Kannagi

Page du projet : lien 63
Description :
Un *shoot them up* pour la Super Nintendo codé en assembleur.
Retour sur l'événement :
Probablement le meilleur week-end de programmation de jeux vidéo du site, beaucoup de participants et j'ai eu de très bons retours sur mon jeu, ça a été une bonne chose pour moi et pour les autres qui ont pu voir un tel jeu se faire en deux jours.

2.12 Le stylo par Antoine, Charlotte, Florie Pa., Vincent, Jérémy, Bastien, Youri, Lionel, Romain L., Cyrielle, Romain B., Marie, Florie Po, Kévin, François, Karine

Page du projet : lien 64
Description :
Le Stylo est un jeu de Texte où le joueur doit faire des choix à des moments de l'histoire qui lui permettront ou pas de finir le jeu et avoir la bonne fin.
Téléchargement : Jeu : lien 65.
Retour sur l'événement :
C'était notre première participation à cet événement, nous étions un groupe composé de pas mal de personnes, mais chacun a beaucoup aimé cette expérience. Nous reviendrons donc l'année prochaine pour renouveler l'aventure.

2.13 SpaceDefend par Yétimothée

Page du projet : lien 66
Description :
Construisez et améliorez des tours pour détruire les cochonneries venant des secteurs voisins de l'espace, et ainsi empêcher le secteur lambda de l'univers d'être envahi (après tout, c'est là que vous vous installerez après votre retraite !).
Téléchargement : Jeu : lien 67. Source : lien 68.
Retour sur l'événement :
Malgré la douloureuse tête dans le coltar de samedi j'ai pu rendre un jeu qui fonctionne à peu près, même si avec assez peu d'intérêt. J'ai été très intéressé par le travail de chacun, presque toujours digne de beaucoup d'intérêt, que ce soit techniquement ou bien « gameplayement » (ou les deux). La seule chose que je reproche, c'est que ça se finit à minuit le dimanche tandis que la plupart bossent le lendemain, ce qui empêche de réunir tout le monde sur le tchat et de tous discuter à propos du week-end ! À mon avis, il serait plus judicieux d'organiser une petite période de tests/critiques à la fin du week-end pour renforcer la convivialité, quitte à perdre un peu de temps

2.14 TetrisCommand par LittleWhite

Page du projet : lien 69

Description :

Je voulais faire un mix entre Tetris et Missile Command, finalement, c'est un remake de Missile Command, simplement.

Téléchargement : Jeu (Windows) : lien 70. Source : lien 71.

Retour sur l'événement :

Quelle bonne surprise de voir autant de monde pour cette édition. Bravo à tous pour vos participations.

2.15 Expendatux par honosstoa

Page du projet : lien 72

Description :

Un mini supertux où les balles tirées ne se comptent pas ! (je le continuerai quand j'aurai du temps libre.)

Retour sur l'événement :

J'ai trouvé ça pas mal de voir autant de monde motivé, même si je regrette un peu de ne pas avoir eu assez de temps libre. Si je peux, je réessayerai l'année prochaine.

2.16 Papi Commando Online par MoDDiB

Page du projet : lien 73

Description :

Version jouable en coopératif online de Papi Commando, avec friendlyfire. Ainsi que l'édition online et coopérative du niveau de jeu. De plus, tout le jeu est modable via des fichiers de configuration en JSON + Lua, il est donc tout à fait possible de rajouter de nouveaux ennemis et de nouvelles armes par exemple ! Le point le plus important pour moi était de tester et d'améliorer le netcode afin de voir si un tel jeu était jouable en ligne de manière agréable.

Téléchargement : Jeu : lien 74.

Retour sur l'événement :

Stimulant ! C'est vraiment génial de pouvoir bosser sans se sentir seul, de se serrer les coudes et de partager nos expériences : à renouveler dès que possible !

2.17 Application web de Black Jack par joltcounter

Page du projet : lien 75

Description :

Un simple jeu de blackjack où vous jouez contre la banque. Le jeu est fait sous forme d'application Web, donc vous n'avez besoin de rien installer. Pour ce qui est des technologies utilisées, ce jeu est fait en HTML5 et utilise la balise canvas, l'API localstorage (pour stocker votre meilleur score) et l'API offline, car oui ce jeu est disponible en mode hors-ligne ! En langage de programmation, j'ai utilisé Dart de Google que j'ai compilé en JavaScript. Ce jeu fonctionne sur les navigateurs récents.

Téléchargement : Version en ligne : lien 76.

Retour sur l'événement :

J'ai bien aimé l'événement, ça m'a permis de faire mon premier jeu de cartes (chose que je voulais faire depuis longtemps). Je participerai certainement à celui de l'année prochaine. Ces 48 h de programmation sont vraiment une bonne façon de se motiver à créer un jeu et c'est agréable de voir qu'en si peu de temps on peut réussir à produire quelque chose de bien et fonctionnel. Cet événement est une bonne façon de se tester.

2.18 Autres projets

Hexagoune par Epitouille et Epilad : lien 77. Galaxy Bird par Cocorico Games : lien 78.

*Retrouvez l'article d'**Alexandre Laurent** en ligne : lien 79*

JavaScript

Les derniers tutoriels et articles

Débat : Les Web Components, pour le meilleur et pour le pire ?

Les Web Components ont fait beaucoup parler d'eux depuis l'avancée des dernières spécifications et le développement de *polyfills* permettant de les utiliser dès maintenant. Ils ont même été décrits par plusieurs articles de presse comme la prochaine révolution du développement web. Mais cet engouement est-il justifié ? Pour aller à contre-courant de la multitude d'articles en vantant les mérites, nous allons détailler dans cet article l'intérêt discutable, les limitations, les inconvénients et les mauvais cas d'utilisation des Web Components.

1 Les promesses des Web Components

1.1 Un standard pour les composants d'interface en JavaScript

La particularité de cette spécification est qu'elle vient influencer la manière dont nous codons nos pages HTML en y ajoutant une approche résolument **modulaire**. Les composants viennent se situer hiérarchiquement entre le document et les éléments ; on peut les concevoir comme des groupes autonomes d'éléments formant un bloc identifiable visuellement et fonctionnellement.

Ainsi, ils se substituent assez bien au concept de **widgets**, ces composants d'interfaces déjà popularisés depuis de nombreuses années par des bibliothèques comme jQueryUI (lien 80), YUI (abandonné il y a peu par Yahoo)(lien 81) ou encore les widgets Dojo (lien 82). Le net avantage des Web Components par rapport à ces bibliothèques est la simplicité d'intégration, puisqu'il suffit d'utiliser la balise personnalisée dans votre HTML pour charger le composant. Les développeurs de composants sont encouragés à permettre le paramétrage de ces composants directement via des attributs HTML, afin que le composant puisse être utilisé sans écrire une ligne de JavaScript. Toutefois, une simple déclaration HTML ne peut égaler la flexibilité du paramétrage et de l'initialisation JavaScript, c'est pourquoi certains Web Components complexes s'utilisent encore par le biais d'une API JavaScript.

Ce nouveau standard devrait donc en théorie sonner le glas de ces différentes bibliothèques UI, ce qui fait que nous n'aurions plus à nous préoccuper des dépendances propres à chaque widget trouvé sur le Net. En théorie seulement, car on trouve déjà de nouvelles bibliothèques regroupant des Web Components avec un format, un style et des spécificités qui leur sont propres : Polymer de Google (lien 83) et Brick de Mozilla (lien 84) par exemple. Pour l'instant, ils font des efforts en matière d'interopérabilité (lien 85), mais nous n'avons aucune garantie que cela restera le cas avec l'apparition d'autres bibliothèques de ce genre. Malheureusement, les bibliothèques de Web Components semblent tomber dans les mêmes travers que les bibliothèques de widgets en leur temps.

1.2 Simplifier et étendre la syntaxe HTML

Une critique récurrente du HTML par les développeurs d'applications web riches est le nombre restreint d'éléments à disposition : des grands absents comme les barres d'onglets, les *tooltips* (infobulles) ou les *overlays* (masques d'arrière-plan) se font toujours désirer. Tout le problème du point de vue des organismes de standardisation est de définir la nature de ces éléments, et de s'assurer qu'ils présentent un sens pour tous les usages et contextes qu'offre le Web. Par exemple, quelle différence y a-t-il entre une barre d'onglets, un menu accordéon, et une barre de navigation classique chargeant du contenu via AJAX ? Certes, chacun dispose de caractéristiques visuelles particulières qui nous permettent de les différencier. Mais sur le plan sémantique et fonctionnel, c'est une autre affaire. C'est pourquoi les descriptions des éléments standards sont volontairement abstraites : <article> ne contient pas forcément un article au sens publication, tandis que <figure> peut convenir à la fois pour des images, des schémas ou des bouts de code.

Puisqu'avec les Web Components, chacun est

libre de créer les balises de son choix, le futur HTML sera sans doute beaucoup plus diversifié et concret. Bonne ou mauvaise nouvelle ? Les développeurs s'en réjouissent au premier abord, mais un HTML trop spécifique et disparate pourrait nuire à l'ubiquité du Web : si les experts du W3C et du WHATWG prennent bien le temps de peser le pour et le contre à l'introduction d'un nouvel élément, il y a peu de chances que nous fassions preuve d'autant de réflexion (surtout un vendredi soir sur un projet en retard Image non disponible). Un élément barre d'onglets <tabs-bar> peut être une fausse bonne idée s'il doit être remanié en menu déroulant sur les petits écrans.

Le Web se transforme sans cesse et ses usages s'étendent à de nouvelles classes d'appareils. Ce phénomène semble s'accentuer avec le temps, ce qui me laisse à croire que la vision que nous avons du Web est encore très limitée : le Web n'a que 25 ans, et il a déjà changé de visage plus d'une fois (Web 2.0, révolution mobile...). Aujourd'hui plus que jamais, nous avons besoin de prendre du recul sur la conception de nos pages Web et de nous concentrer sur le contenu, à travers un HTML générique et sémantique. Un HTML à l'abstraction et aux limites nécessaires, qui ne parte pas dans toutes les directions et qui ne se banalise pas, car son rôle est bien plus important qu'on ne le croit.

Certains font preuve d'un enthousiasme exagéré vis-à-vis des Web Components... Source : lien 86

1.3 Standardiser le templating côté client

L'introduction des Web Components amène également la balise <template>, dont la fonction est très particulière : elle contient un fragment de HTML qui sera chargé sans être activé : c'est-à-dire que tous les éléments à l'intérieur sont inertes avant d'être utilisés : les images ne sont pas chargées, les <script> ne sont pas exécutés, les <video> et <audio> ne sont ni chargés ni joués... Ce HTML inerte peut par la suite être « activé » en copiant son contenu et en l'insérant en JavaScript à l'endroit désiré dans le document.

Cette nouvelle balise est pressentie pour standardiser le templating côté client, comme le souligne cet article sur HTML5Rocks.com : lien 87. Or, il n'en est rien. Comme nous le verrons en section IV de cet article, cet objectif est bien trop large et abstrait pour être atteignable. Les auteurs de la spécification en ont bien conscience et n'ont pas même pas essayé d'aller dans cette voie. En réalité, la spécification est très limitée dans son champ d'application. Il s'agit simplement d'un conteneur de HTML à réserver pour plus tard, ce que beaucoup de bibliothèques de templating client avaient déjà réussi à faire par le biais de balises <script> avec un attribut type spécial.

Tous les autres usages envisagés, et en particulier son rôle de pierre angulaire dans les solutions de templating côté client, ne sont que pure extrapolation de la part des développeurs. Actuellement, la spécification présente plus de contraintes que d'avantages comparée aux solutions existantes à base de <script>. Elle a le mérite d'enfin apporter un standard pour un mécanisme utilisé et éprouvé depuis des années. Mais tout comme les imports HTML, elle semble arriver trop tard et les éléments <template> seuls ne servent à rien sans JavaScript pour les activer.

2 Custom Elements : <mon-element-a-moi>

2.1 Adieu les définitions de type de document (DTD)

HTML, un descendant de SGML (Standard Generalized Markup Language), a pu jusqu'à sa version 4 être décrit par des DTD, c'est-à-dire des descriptions techniques formelles du langage qui faisaient notamment la liste des balises admises dans le langage.

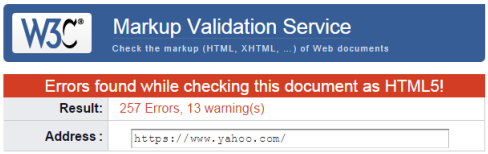
```
1  <!ELEMENT IMG - O EMPTY                -
       - Embedded image -->
2  <!ATTLIST IMG
3    %attrs;                              -- %
         coreattrs, %i18n, %events --
4    src        %URI;     #REQUIRED -- URI of
          image to embed --
5    alt        %Text;    #REQUIRED -- short
         description --
6    longdesc   %URI;     #IMPLIED  -- link
         to long description (complements
         alt) --
7    name       CDATA     #IMPLIED  -- name
         of image for scripting --
8    height     %Length;  #IMPLIED  --
         override height --
9    width      %Length;  #IMPLIED  --
         override width --
10   usemap     %URI;     #IMPLIED  -- use
         client-side image map --
11   ismap      (ismap)   #IMPLIED  -- use
         server-side image map --
12   >
```

Grâce aux DTD, les éléments de la page sont systématiquement décrits et normalisés. Ces éléments connus, les navigateurs disposent d'informations pour déduire la hiérarchie du document et le

représenter plus efficacement.

Parmi les cousins les plus connus du HTML, XML offre bien plus naturellement aux développeurs la possibilité d'utiliser leurs propres balises pour décrire les données ; les DTD qui les accompagnent font office de contrat d'interface très fiable, faisant du XML un bon choix pour la communication entre serveurs et web services.

HTML a pris un autre chemin. Les interpréteurs dans les navigateurs sont conçus pour être très tolérants, en acceptant les erreurs de syntaxe et en tentant de les corriger : c'est pourquoi la plupart des navigateurs web parviennent à fermer automatiquement certaines balises, à afficher des balises inconnues ou à ignorer les attributs et valeurs non standardisées. De ce fait, les développeurs Web ont toujours fait preuve de beaucoup de négligence sur la qualité et la validité stricte du HTML qu'ils produisaient. Si vous avez déjà utilisé un validateur HTML (on en trouve des dizaines gratuits en ligne, dont celui du W3C : lien 88), vous avez sans doute déjà essayé de faire valider certains sites grand public et le nombre d'erreurs relevées a dû vous surprendre.



Ainsi, les développeurs HTML n'ont jamais vraiment porté grande importance aux DTD. Depuis HTML5, les éléments n'ont d'ailleurs plus à respecter les contraintes d'une DTD ; ce que les développeurs ont très bien accueilli par le simple fait que cela leur simplifiait la déclaration du DOCTYPE :

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
      4.01 Transitional//EN"
2   "http://www.w3.org/TR/html4/loose.dtd
        ">
```

```
1 <!DOCTYPE html>
```

Avec les *Custom Elements* (éléments personnalisés), cette flexibilité va encore plus loin puisque de nouveaux types d'éléments peuvent être définis côté client a posteriori de la création du document ; ce que n'a jamais permis une DTD. La majorité des articles autour des Custom Elements ne font d'ailleurs aucune mention des DTD, ce que je trouve très étonnant. Du temps du HTML4, si on avait demandé à un développeur de déclarer des balises personnalisées, il aurait tout de suite pensé à une DTD personnalisée.

Malgré le fait que le HTML5 se soit émancipé des DTD, on pourrait tout de même en parler ne serait-ce qu'à titre de comparaison.

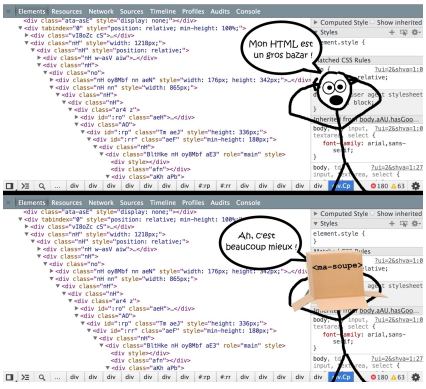
2.2 Shadow DOM : le cache-misère officiel

Les Custom Elements sont composés d'éléments HTML standards. Ce ne sont pas des nouvelles briques, mais un assemblage de briques existantes agrémentées de JavaScript et de déclarations CSS isolées. Il est important de comprendre que tout ce que l'on peut faire avec les Custom Elements peut déjà être fait aujourd'hui : **il s'agit simplement d'un emballage, d'un paquet cadeau à destination des développeurs**, destiné à normaliser la définition et l'intégration d'éléments d'interface aussi appelés widgets, bien que le terme semble être passé de mode.

Puisque tout l'intérêt des Custom Elements est d'avoir un bel emballage, il faut veiller à ce que le contenu de ces éléments soit bien emballé et caché d'un regard extérieur tant que l'on n'a pas ouvert la boîte : c'est de là que vient l'idée du Shadow DOM.

Mais que cherche-t-on à cacher exactement ? Qu'y a-t-il de si affreux et verbeux dans notre HTML pour que l'on souhaite le faire disparaître ? Tous les exemples donnés dans les articles sur ce sujet sont criants de vérité : c'est l'utilisation abusive de markup à des fins de mise en forme.

En effet, il semble tout à fait normal pour certains auteurs d'articles sur le sujet (lien 89) que l'on distingue en HTML un markup de présentation d'un markup de contenu. Le markup pour le style, ce n'est pas sans rappeler la mise en page par tableaux qui avait ses émules entre 1995 et 2005 : utiliser des éléments <table>, <tr> et <td> pour aligner et dimensionner les éléments au mépris de la sémantique du document. Si la mise en page par tableaux est aujourd'hui considérée comme une mauvaise pratique (et même source de moqueries pour certains web designer seniors), force est de constater que l'utilisation abusive de et de , aussi appelée « soupe de div », ne fait guère mieux. On comprend mieux pourquoi on souhaite la cacher.



Le Shadow DOM, ou la politique de l'autruche

Voici un extrait des Web Components Best Practices (lien 90) issu du site communautaire webcomponents.org (lien 91) :

> Don't put too much in Shadow DOM : Shadow DOM allows you to stuff

> a bunch of complex junk out of sight. However, that's not an excuse to have as many DOM elements as you want in your shadow, as more elements will still lead to worse performance. In addition, try to keep your configuration and state visible by keeping anything semantic exposed in the logical DOM. Cruft goes in the Shadow ; semantic stuff doesn't.

Traduction : Ne chargez pas trop votre Shadow DOM : Le Shadow DOM vous autorise à fourrer votre camelote à l'abri des regards indiscrets. Cependant, ce n'est pas une excuse pour avoir autant d'éléments DOM que vous avez envie dans votre Shadow DOM, car un nombre croissant d'éléments conduira immanquablement à empirer les performances. De plus, essayez de garder votre configuration et vos états applicatifs visibles en maintenant tout ce qui relève de la sémantique dans le DOM logique. Vos cochonneries vont dans le Shadow DOM, les éléments sémantiques non.

S'agit-il là de l'aveu qu'un HTML purement sémantique est impossible ? Ou juste trop difficile à atteindre ? Notez les mots employés : *junk* (camelote) et *Cruft* (cochonneries). Doit-on vraiment accepter cela et normaliser un HTML camelote ?

Mon opinion est qu'il ne devrait pas y avoir du tout de markup de présentation. D'ailleurs, toutes les balises dites de présentation telles que , <font> ou <b> ont été dépréciées ou adaptées en HTML5. S'il reste certains cas nécessitant de modifier le markup uniquement à des fins de style, alors les standards doivent évoluer pour régler ces cas au lieu de les tolérer et de chercher à les dissimuler.

Le Shadow DOM est un cache-misère. Certains diront qu'il s'agit d'une solution pragmatique et court-termiste à ce problème. Je pense personnellement que c'est tout sauf une solution : il ne résout pas le problème du HTML trop verbeux, mais se contente de cacher la poussière sous le tapis. Au contraire, en masquant le problème, cela pourrait encourager les développeurs à persister dans cette mauvaise pratique et ne faire qu'empirer les choses. D'autres diront qu'il présente un intérêt pour l'isolation des règles CSS. Je leur répondrai qu'il existe d'autres moyens pour y parvenir sans avoir à fragmenter le document : <style scoped> @import "composant.css" ; </style>

HTML5 a amené une vingtaine de nouvelles balises liées à la structure et à la sémantique du document, tandis que les évolutions de CSS réduisent peu à peu la nécessité de recourir à de multiples éléments à des fins de mise en forme. Il s'agit clairement de la direction à suivre, en s'attaquant au problème de front. C'est pourquoi le Shadow DOM doit rester à sa place : à l'ombre !

2.3 Pas si custom que ça

Avec les Custom Elements, on s'autorise à ce que le HTML parte dans toutes les directions à partir du moment où les propriétés de ces nouveaux éléments accompagnent le document : est-ce réellement une bonne idée ? Outre le fait de charger un peu plus encore les requêtes, on vient surcharger les propriétés établies par défaut par le navigateur pour chaque élément.

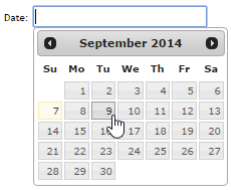
En effet, chaque navigateur implémente sa propre feuille de style, appelée *User Agent Stylesheet*, qui définit les propriétés CSS de base des éléments des pages que vous consultez. Ce sont ces règles que vous surchargez en définissant vos propres feuilles de style pour vos sites. Or, les User Agent Stylesheets varient selon le navigateur ; c'est ce qui explique que vous n'ayez pas tout à fait le même rendu d'un navigateur à un autre (ça et quelques détails d'implémentation des moteurs de rendu).

Sur un même navigateur, elles peuvent varier également ; par l'ajout d'extensions, de feuilles de styles propres à l'utilisateur (*userstylesheets*), de détails spécifiques au terminal (par exemple un navigateur Android selon s'il est installé sur un smartphone ou une tablette), ou encore d'outils dédiés à certains handicaps : ces outils peuvent agrandir les textes, augmenter le contraste, fournir des moyens de saisie et de navigation assistés, simplifier la mise en page, etc.

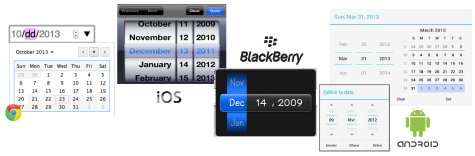
Ces variations sont utiles et nécessaires : le développeur ne devrait en aucune façon chercher à les annihiler pour que son site s'affiche de la même manière sur tous les navigateurs. Or, plus on définit les propriétés d'éléments, moins les règles par défaut s'appliquent. Ce contrôle absolu de l'apparence et du comportement du composant peut également poser des problèmes d'accessibilité. Heureusement, l'accessibilité n'a pas été oubliée par tous les développeurs de Web Components et fait déjà partie des préoccupations des auteurs de Polymer (lien 92) ; il faudrait idéalement qu'elle ne soit oubliée par personne.

Pour mieux comprendre quel est le problème exactement, prenons un exemple parlant et intéressons-nous un moment au cas du sélecteur de dates (*datepicker*). C'est un cas que j'ai rencontré plusieurs fois et qui m'a vraiment aidé à comprendre les enjeux de la standardisation d'éléments.

En HTML4, les boîtes de saisie sont très limitées dans les formulaires et le seul moyen de sélectionner une date est de la saisir dans un certain format : DD/MM/YYYY par exemple. Une validation de format est ensuite effectuée côté serveur, et parfois côté client également. Tout cela est assez fastidieux à la fois pour l'utilisateur comme pour le développeur. Ainsi, le widget DatePicker est rapidement devenu un classique des bibliothèques de widget. Voilà à quoi ressemble celui de jQueryUI :



Ensuite, avec HTML5 est apparu <input type="date">. Maintenant que le type de donnée attendu est connu des navigateurs, ceux-ci peuvent proposer leurs propres datepicker conçus spécifiquement pour chaque terminal et navigateur ; ce qui se traduit par un moyen de saisie plus adapté et ergonomique. Opera a été parmi les premiers à implémenter un datepicker ; ils sont ensuite apparus sur divers navigateurs mobiles. Voilà un aperçu de certains datepicker « natifs » existants :



Datepickers natifs sur Chrome, iOS, BlackBerry et Android

Notez la différence d'ergonomie entre un datepicker pour souris comme celui de Chrome et un datepicker pour écran tactile comme sur iOS ; ou la différence entre les datepicker Android selon la taille d'écran. Les datepicker JavaScript ne pourront jamais s'adapter aussi bien selon le contexte ; de plus, les utilisateurs sont accoutumés à utiliser les composants natifs fournis par leur navigateur et leur système : ils sauront l'utiliser sans problème.

Certes, tous les navigateurs ne proposent pas encore de widgets : Firefox et Internet Explorer considèrent encore ces types de champ comme de simples champs texte. Mais il est possible de détecter le support de datepicker natif et de charger un datepicker JavaScript comme solution de secours (un tutoriel est disponible ici : lien 93). Le widget pourrait faire lui-même ce test, ou au moins proposer un paramétrage à cet effet. Cependant, ce n'est généralement pas l'approche choisie par les développeurs de ces widgets : ils préfèrent choisir très précisément le style et le comportement de leur widget, quitte à négliger l'adaptation au terminal.

Et c'est le même topo pour les Web Components : alors que la spécification est toujours à l'étude, on voit déjà plusieurs composants sélecteurs de date faire leur apparition, dont la grande majorité ne se préoccupent pas de tester le support de <input type="date">. Il y a de rares exceptions comme celui-ci qui agit comme un polyfill : lien 94. Mais pourquoi alors utiliser un Custom Element quand un polyfill sur l'élément input fonctionne tout aussi bien ?

En résumé, un des risques majeurs liés à l'usage excessif de Custom Elements est celui de surcharger excessivement les styles par défaut d'éléments standards ou de ne pas utiliser les éléments standards appropriés, ce qui nuit à l'ergonomie et l'accessibilité de vos composants.

3 HTML imports : arrivés après la bataille

3.1 Alors qu'on ne les attendait plus

Nous composons nos pages HTML de manière modulaire depuis de nombreuses années. L'inclusion de HTML est une des fonctionnalités les plus basiques de tout langage de vues côté serveur ; elle préfigure souvent dans les tutoriels. Ainsi, ces lignes de code ne devraient pas vous être inconnues :

```
1 <?php include 'header.php'; ?>
2
3 <jsp:include page="header.jsp" />
4
5 <!-- #include file="header.asp" -->
```

Sans langage serveur, c'est-à-dire avec un serveur web statique, il existe également plusieurs moyens d'inclure une page dans une autre. Lorsque j'ai fait mes débuts en développement web il y a douze ans, les framesets étaient à la mode :

```
1 <frameset rows="20%,80%">
2    <frame name="header" src="header.html
         " />
3    <frame name="main" src="main.html" />
4    <noframes>Your browser does not
         support frames.</noframes>
5 </frameset>
```

Ensuite on a voulu apporter une solution moins intrusive et plus discrète : les iframes

```
1 <iframe src="header.html"></iframe>
```

Puis on s'est aperçus qu'en bricolant un peu, on pouvait se débrouiller avec JavaScript pour composer nos vues tout en conservant un seul document :

```
1 <script src="header.html.js"></script>
```

```
1 document.write("<h1>contenu échappé de
      header.html ici</h1>");
```

Enfin, AJAX est arrivé et tout est devenu beaucoup plus facile :

```
1 <script>
2   function include(src){
3     var scriptRef = document.scripts[
          document.scripts.length-1];
4     var xhr = new XMLHttpRequest();
5     xhr.onreadystatechange = function(){
6       if (xhr.readyState==4 && xhr.
            status==200){
7         scriptRef.outerHTML = xhr.
              responseText;
```

```
      }
    };
    xhr.open("GET", src, true);
    xhr.send();
  }
</script>
</head>
<body>
<script>include("header.html");</script>
```

Nous avons donc aujourd'hui tout un panel de solutions éprouvées à disposition. La dernière en date, AJAX, est utilisée couramment depuis 2006, soit depuis plus de huit ans. Et voilà qu'en 2014 arrive timidement une spécification pour les imports HTML. Encore au statut de Working Draft au W3C, on peut espérer dans le meilleur des cas l'utiliser à la mi-2015 avec un polyfill AJAX comme parachute. Mais qu'avons-nous à y gagner à utiliser cette nouvelle spécification par rapport à AJAX ou de la composition côté serveur ? Les solutions actuelles basées sur AJAX sont plus flexibles et leur large support jouera en leur faveur encore plusieurs années.

3.2 Un import seul ne sert à rien

Les imports HTML ne font pas vraiment ce qu'ils sont supposés faire au premier abord : ils n'ajoutent aucun HTML directement dans votre document. Ils se contentent de charger le document HTML afin que vous puissiez l'exploiter ensuite en JavaScript. Est-ce que vous vous attendiez à cela ? Alors que toutes les instructions d'import dans les langages serveur comme PHP, JSP ou ASP chargent le contenu et l'insère à l'endroit de l'instruction, un import HTML ne fait rien du tout avec le contenu. C'est totalement contre-intuitif et ne suit pas les autres comportements de la balise <link>. Imaginez que les feuilles de styles CSS importées ne s'appliquent pas par défaut sur le document, et qu'il faille les activer manuellement en JavaScript.

Par ailleurs, cela rend caduc l'avantage supposé de cette solution comparée à celles qu'on utilisait jusque-là : l'import de HTML par balises <script> ou par AJAX nécessite également un recours au JavaScript, avec un code de taille similaire :

```
function include_AJAX(src){
  var scriptRef = document.scripts[
      document.scripts.length-1];
  var xhr = new XMLHttpRequest();
  xhr.onreadystatechange = function(){
    if (xhr.readyState==4 && xhr.status
        =200){
      scriptRef.outerHTML = xhr.
          responseText;
    }
  };
  xhr.open("GET", src, true);
  xhr.send();
}

function include_import(src){
  var scriptRef = document.scripts[
      document.scripts.length-1];
  var link = document.createElement('
      link');
  link.rel = 'import';
  link.onload = function(e) {
    var body = link.import.querySelector
        ("body").cloneNode(true);
    scriptRef.outerHTML = body.innerHTML
        ;
  };
  link.href = src;
  document.head.appendChild(link);
}
```

Ce qui amène des incohérences assez amusantes, par exemple dans l'article de HTML5Rocks (lien 95) :

> AJAX
>
> I love xhr.responseType="document", but you're saying I need JS to load HTML ? That doesn't seem right. J'adore xhr.responseType="document", mais vous dites que j'ai besoin de JavaScript pour charger du HTML ? Ça ne semble pas être l'idéal.

Et plus loin...

> Including an import on a page doesn't mean "plop the content of that file here". It means "parser, go off an fetch this document so I can use it". To actually use the content, you have to take action and write script. Inclure un import dans une page ne signifie pas "mets le contenu de ce fichier ici". Cela signifie : "parseur, va charger ce document afin que je puisse l'utiliser". Pour utiliser concrètement le contenu, vous devez agir dessus en JavaScript.

Il est évident que l'on importe du HTML pour se servir de ce contenu : alors, d'où vient cette drôle d'idée de ne pas pouvoir se servir du contenu sans recourir au JavaScript ? Justement du fait qu'ils arrivent trop tard : les solutions complexes mises en place dans les frameworks JavaScript pour gérer les vues et sous-vues proposent un panel de fonctionnalités que HTML seul ne pourra jamais rivaliser avec. L'utilisation de JavaScript est une trappe de sortie pour la spécification qui lui permet d'esquiver sa pauvreté apparente et ne pas devoir faire face à un constat amer : celui qu'elle arrive avec quinze ans de retard.

4 La balise <template> : la coquille vide qui veut régner sur l'océan

4.1 Mille solutions à un seul problème

Il y a un an, je publiais ici un article décrivant tout ce qu'il y a à savoir sur le templating côté client : lien 96. J'ai écrit cet article en m'imposant une consigne : celle de ne pas privilégier une approche plutôt qu'une autre, mais au contraire de toutes les parcourir, les illustrer par l'exemple et les comparer.

En explorant les différentes approches, je me les suis représentées en les positionnant sur un axe simple : cet axe va de « Full Logic » à « Logic-less » selon la quantité de logique présente dans le template. Mais les différences vont bien au-delà de ça. Tout comme on ne peut pas réduire la politique à la gauche ou la droite, on ne peut pas résumer une solution de templating à la richesse syntaxique de ses templates. Le temps de rendu, la précompilation, le préprocessing, les méthodes de rendu partiel, l'asynchronicité du rendu ou encore l'empreinte sur le DOM sont autant de critères à prendre en compte.

La conclusion de cet article sur le templating côté client était sans appel : il existe des tas de solutions différentes avec des approches différentes. Dès lors, comment une seule spécification, ou devrais-je dire un seul élément, parviendrait-il à remplacer ce panorama de solutions ?

4.2 Parce qu'il faut les mettre dans une case

La réponse à la question précédente est très simple : il n'y parviendra pas. Et il n'a jamais été question dans la spécification de ne serait-ce qu'essayer d'arriver à la cheville d'une solution de templating existante aujourd'hui. Je pense que l'introduction de la balise <template> répond bêtement au besoin de mettre le templating client dans une case ; comme s'il fallait sous-entendre son existence d'une quelconque manière dans une section de la spécification HTML.

Une case ou plutôt une boîte noire. En effet, pour essayer de convenir au plus grand nombre de méthodes de templating, les auteurs de la spécification ont voulu que le contenu HTML dans cet élément soit totalement inerte. Ainsi, il n'interfère pas avec le reste du document tant qu'il n'a pas été activé en JavaScript ; on laissera le soin à chaque bibliothèque de templating de l'activer au moment désiré et de la manière qui lui sied.

En nommant cet élément <template>, on peut légitimement dire qu'il y a tromperie sur la marchandise. Ne cherchez pas comment vous pouvez vous servir de cet élément pour concrètement écrire des templates. **Cet élément ne fait rien de base, même pas d'interpolation de données !** Aucune instruction logique, aucune référence à un modèle de données. Rien de tout ça. C'est une boîte vide.

En réalité, cet élément est là uniquement pour remplacer le point d'ancrage des templates des solutions existantes qui est généralement une balise <script> avec un type inconnu du navigateur pour l'empêcher de tenter de l'interpréter :

```
1  <script id="mon-template" type="text/x-
       handlebars-template">
```

Il ne remplace donc absolument pas toutes les bibliothèques de templating existantes, contrairement à ce que le nom de l'élément pourrait laisser supposer. Cela a laissé perplexe beaucoup de gens, et certains participants à l'élaboration de cette spécification regrettent ce nommage :

Traduction : j'ai le sentiment que nous aurions dû appeler cet élément <inert> au lieu de <template>.

4.3 Une utopie de standard

Pour essayer de convenir à toutes les approches, la spécification a opté pour en faire le moins possible ; à vrai dire presque rien. Mais le peu qu'elle fait sort déjà du cadre de certaines solutions de templating, notamment celles qui viennent s'interfacer sur des éléments du DOM déjà actifs dans le document avant le rendu du template. On peut aussi se demander comment cette spécification va permettre de gérer les états, les saisies formulaire, la compression ou précompilation des templates, les instructions logiques avancées qui influent sur la structure du DOM... Autant de questions déjà traitées par les bibliothèques de templating, mais auxquelles cette spécification n'apporte aucune réponse.

Le rafraichissement partiel de templates pose également problème. Prenons l'exemple d'un template qui affiche une liste d'articles. À la suite de l'ajout d'un article, nous souhaitons actualiser la vue en insérant uniquement l'élément du nouvel article, sans toucher aux autres. Or la référence logique à la boucle se trouve en principe dans le template, et non dans le DOM déjà généré. Il faut donc soit employer une méthode alternative de rendu pour les rafraichissements partiels, soit travailler sur un mécanisme de comparaison pour actualiser uniquement les parties du DOM ayant changé. Ce mécanisme, aussi appelé **dirty-checking**, est utilisé par plusieurs solutions

complexes (notamment AngularJS, mais s'avère très consommateur de ressources.

De manière plus générale, il semble que le concept de templates statiques déclarés individuellement soit déjà dépassé comparé aux approches modernes dites de **data-binding**. On ne travaille plus avec des templates isolés, mais avec un ensemble de liens données/vue permettant un rafraîchissement intelligent et autonome. C'est pourquoi je doute qu'AngularJS évolue pour reposer sur cet élément <template> qui s'intègre assez mal avec le fonctionnement existant. Je n'irais pas jusqu'à dire que cette spécification est mort-née, mais son usage est certainement bien plus minoritaire qu'elle ne l'a laissé supposer.

5 Des mauvaises utilisations des Web Components

Suite à tous les défauts évoqués, parcourons divers exemples existants qui les mettent en évidence :

5.1 Le Web Component minimaliste

Un Web Component dont le code est bien trop simple pour justifier sa déclaration comme composant, ou qui ne s'utilise qu'une seule fois.

Exemples :

lien 97

lien 98

5.2 Le Web-Component tout-en-un

Quand on s'amuse à mettre tout le contenu de son site web dans un seul élément *parce que c'est cool*, ou que l'on cherche à mettre des Web Components à l'intérieur d'un Web Component...

Exemples :

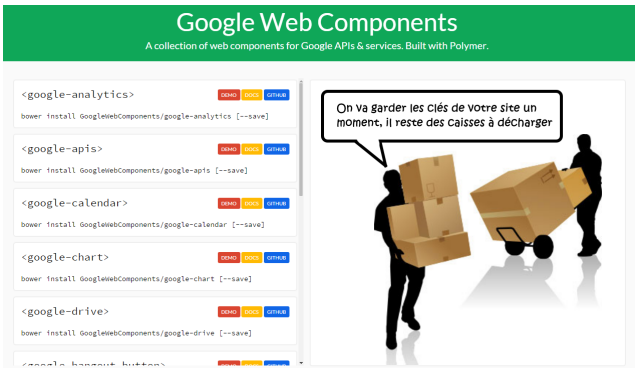
lien 99

5.3 Le Web Component boîte noire ou iframe 2.0

Un Web Component que l'on pourrait aisément remplacer par un iframe tout en réduisant le risque de failles de sécurité. Souvent, il s'agit d'un composant qui charge de multiples scripts externes, avec un code difficile à analyser, très peu de contrôle sur le code importé et pouvant contenir d'éventuelles failles XSS. Il peut aussi s'agir d'un service Web notoire qui vient s'installer de manière un peu trop intrusive sur votre site web en chargeant de grosses API propres au fonctionnement du service (ce que les entreprises appellent parfois « SDK JavaScript », quoi que cela veuille dire).

Exemple :

lien 100



Si vous utilisez un Web Component externe, pensez toujours à regarder dans la boîte.

5.4 Le Web Component anti-standard

Un Web-Component qui reprend le fonctionnel d'un standard existant sans utiliser ce standard, ou qui pourrait être remplacé par un polyfill.

Exemples :

lien 101

lien 102

5.5 Le Web Component qui n'a rien à faire dans HTML

Un Web Component qui ne représente pas d'élément, qui n'est pas identifiable visuellement ou qui ne présente aucun sens sémantique dans le document et peut être remplacé par un script explicite.

Exemple :

lien 103

5.6 Le Web Component gadget

Un Web Component qui n'a aucune raison d'exister, mais qui existe ; pour le fun, pour la publicité ou simplement parce qu'on peut le faire. Le point préoccupant est que ces gadgets se retrouvent au même plan que d'autres composants « sérieux » dans les sites les référençant. Cela amène la question de comment s'assurer de la qualité et de la pérennité des composants trouvés sur le Net.

Exemples :

lien 104

lien 105

6 Conclusion

J'ai utilisé l'humour et un ton assez provocateur dans cet article pour mettre en exergue les défauts des Web Components. Même si certains aspects de la spécification sont à revoir complètement selon moi, tout n'est pas à jeter. Je pense que tous les articles et vidéos déjà parus sur le sujet ont bien expliqué en large et en travers les avantages des Web Components, c'est pourquoi je me suis concentré sur le négatif ici.

Les Web Components, lorsqu'ils sont correctement utilisés, apportent une meilleure lisibilité dans les pages HTML comportant beaucoup d'éléments d'interface sophistiqués (ce qu'on pourrait typiquement appeler les web-apps). Ils permettent de modulariser le HTML, de faciliter la maintenance du code et de réduire le risque d'effets de bord et de régressions fonctionnelles grâce à l'isolation apportée par les fragments de document.

Toutefois, il n'y a pas d'alternative avec les éléments standardisés du HTML. Bien qu'il s'utilise au même niveau qu'un élément HTML, un Web Component est un enrobage autour d'un ou plusieurs éléments accompagnés de CSS et de JavaScript. Il vaut toujours mieux privilégier l'élément standard adéquat plutôt que de vouloir redéfinir les propriétés et le comportement d'éléments neutres tels que ou pour arriver à l'objectif désiré (rappelez-vous l'exemple du datepicker). En effet, les propriétés du composant se limitent au périmètre envisagé par son développeur, contrairement aux éléments standards qui sont adaptés nativement selon le navigateur et le terminal. Ces adaptations ont déjà montré leur utilité par le passé, notamment pour les navigateurs mobiles des premiers smartphones. Il est très difficile (impossible ?) pour un développeur de couvrir tous les contextes d'utilisation existants aujourd'hui, sans même envisager ceux de demain. Et si on y consacrait plus d'efforts, la taille du code des composants en accuserait le coup.

Ainsi, pour savoir s'il est légitime de définir un Web Component pour un besoin donné, voilà ce qu'un « bon » Web Component doit présenter comme caractéristiques selon moi :

- Il n'existe aucun élément standard ou en cours de standardisation qui puisse répondre à ce besoin.
- Il remplit bien la fonction d'élément : il peut être identifié visuellement et placé dans la hiérarchie du document le contenant.
- Il est paramétrable dans une certaine mesure : s'il devient trop complexe, il est sans doute décomposable en plusieurs composants.
- Il est réutilisable, y compris à plusieurs endroits d'un même document.
- Il ne dépend pas du HTML parent dans lequel il se trouve.
- Il est autonome (pas de chargement de scripts externes).
- Il est adapté à un maximum de terminaux (tailles d'écran, interfaces souris/tactile, etc.).
- Il est adapté à un maximum d'utilisateurs (efforts sur l'accessibilité et le contexte d'utilisation).
- Il ne laisse pas de traces mémoire JavaScript une fois l'élément supprimé.
- Son code est lisible et ne laisse planer aucun doute sur son fonctionnement interne.

La publication et le référencement d'un catalogue de web components est une idée très séduisante aux yeux de nombreux développeurs, et plusieurs sites sont déjà à l'œuvre : lien 106, lien 107). Néanmoins, il convient de prendre garde à la qualité et à la pertinence de ces composants : faites le tour des différents composants déjà proposés dans ces catalogues et vous vous apercevrez que bien peu d'entre eux respectent toutes les caractéristiques précédemment listées.

En résumé, les Web Components sont un progrès, car ils normalisent la définition et l'intégration de composants d'interface, sans toutefois révolutionner leur nature elle-même. Cependant, chaque progrès doit être utilisé intelligemment et modérément. Les composants sont des boîtes noires qui peuvent dissimuler un code de mauvaise qualité, des risques de sécurité, des défauts d'accessibilité ou des actions trop intrusives sur le reste de la page. Les sites de référencement des Web Components devront ainsi privilégier la qualité sur la quantité si l'on souhaite que le Web soit bâti avec des briques solides.

Retrouvez l'article de ***Sylvain Pollet-Villard*** *en ligne : lien 108*

Reseau

Les derniers tutoriels et articles

Introduction à l'analyse réseau avec Wireshark

L'analyseur de trafic est un outil pédagogique essentiel pour comprendre les mécanismes de fonctionnement des protocoles de communication sur les réseaux contemporains. Ce document comprend deux parties. Dans un premier temps, on trouve une introduction à l'utilisation de l'analyseur **Wireshark**, le logiciel libre incontournable en la matière. Dans un deuxième temps, les travaux pratiques permettent de découvrir la richesse des informations fournies par cet analyseur.

1 Copyright et Licence

1.1 Métainformation

Cet article est écrit avec DocBook XML (hrefhttp ://www.docbook.org/lien 108) sur un système Debian GNU/Linux (lien 109). Il est disponible en version imprimable au format PDF : lien 110.

2 Analyse avec Wireshark

Avec Wireshark (lien 111), il est possible de capturer des paquets directement sur les interfaces du système utilisé ou de lire des fichiers de captures sauvegardées. **Wireshark** supporte les formats de fichiers de capture les plus courants : libpcap/tcpdump, Sun's snoop/atmsnoop, LanAlyzer, MS Network Monitor, HPUX nettl, AIX iptrace, Cisco Secure IDS, etc.

2.1 Quels sont les protocoles supportés ?

La liste des protocoles supportés par **Wireshark** est considérable. Elle évolue avec chaque nouvelle version. La page Protocol Reference (lien 112) fournit un classement par famille de tous les protocoles dont les champs sont interprétés. Il est aussi possible d'accéder à ces informations via le menu Help ? Supported Protocols.

2.2 Quels sont les médias supportés ?

Le logiciel **Wireshark** permet l'analyse de transmissions réseau sur presque toutes les technologies. Les limitations d'utilisation sont plutôt dues au système d'exploitation sur lequel on exécute ce logiciel. Pour obtenir un état des possibilités d'analyse en fonction du système utilisé, il faut consulter la page : Network media specific capturing (lien 113).

2.3 Comment accéder aux interfaces ?

Lorsque l'on exécute **wireshark** en tant qu'utilisateur normal, on ne peut accéder à la liste des interfaces en lançant l'opération Capture : lien 114. Sur un système d'exploitation correctement administré, un utilisateur normal ne doit pas avoir accès aux interfaces sans condition. Il existe plusieurs solutions pour donner un accès direct à la liste des interfaces physiques. En voici trois.

En mode super-utilisateur

Partant d'une connexion avec un compte utilisateur normal, celui-ci est propriétaire exclusif de son écran (**display**). Il doit donc autoriser le super utilisateur à accéder à son écran à l'aide de la commande **xhost**, passer en connexion super-utilisateur avec la commande **su** puis exécuter l'application **Wireshark**.

```
$ xhost +local:
$ su
Password:
# wireshark &
```

En mode utilisateur avec gksu

L'application gksu est un frontal graphique de la commande **su**. Elle permet, après la saisie du mot de passe super-utilisateur, d'exécuter une application en mode super-utilisateur. Dans notre cas, on accède directement aux interfaces physiques en mode graphique sans passer par une manipulation à la console. Ce mode opératoire est proposé par défaut avec toutes les distributions GNU/Linux actuelles.

En mode utilisateur avec sudo

L'application sudo permet de déléguer les droits du super-utilisateur avec une granularité très fine. L'optique de l'analyse réseau étant un cas particulier de l'administration système, on se limitera à la présentation du fichier de configuration de l'application : /etc/sudoers.

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
#

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) ALL

etu    ALL = NOPASSWD: /usr/bin/wireshark, /usr/bin/tshark
```

C'est à la dernière ligne que se situe la partie intéressante. L'utilisateur normal etu dispose, sur n'importe quel hôte géré par ce système (ALL), d'un accès super-utilisateur aux applications **Wireshark** et tshark sans avoir à saisir son mot de passe. Pour lancer l'application, il faut préciser l'appel à l'application sudo de la façon suivante :

```
$ sudo wireshark &
```

En mode utilisateur via les Linux Capabilities

On débute par la création d'un groupe système dédié à la capture de trafic réseau.

```
# addgroup --system pcap
Adding group 'pcap' (GID 136) ...
Done.
```

On ajoute un ou plusieurs utilisateur(s) au groupe système.

```
# adduser phil pcap
Adding user 'phil' to group 'pcap' ...
Adding user phil to group pcap
Done.
```

Attention ! Cette nouvelle attribution n'est valable qu'après une nouvelle authentification. Nous sommes encore dans le cas classique de création du contexte de travail utilisateur au moment de l'authentification.

On modifie les propriétés du programme dumpcap qui est chargé de la collecte du trafic réseau.

Avant modification du groupe propriétaire, le masque des permissions est le suivant :

```
# ls -lh `which dumpcap`
-rwxr-xr-x 1 root root 62K  4 mars  18:04 /usr/bin/dumpcap
```

On change le groupe propriétaire et on applique un nouveau masque de permissions. Une fois cette opération faite, les membres du groupe système pcap seront les seuls utilisateurs à pouvoir exécuter le programme en mode non privilégié.

```
# chgrp pcap /usr/bin/dumpcap
# chmod 750 /usr/bin/dumpcap
# ls -lh /usr/bin/dumpcap
-rwxr-x--- 1 root pcap 62K  4 mars  18:04 /usr/bin/dumpcap
```

On indique au gestionnaire de paquets Debian que ces nouvelles propriétés doivent être conservées lors des mises à jour à venir.

```
# dpkg-statoverride --add root pcap 750 /usr/bin/dumpcap
# dpkg-statoverride --list /usr/bin/dumpcap
root pcap 750 /usr/bin/dumpcap
```

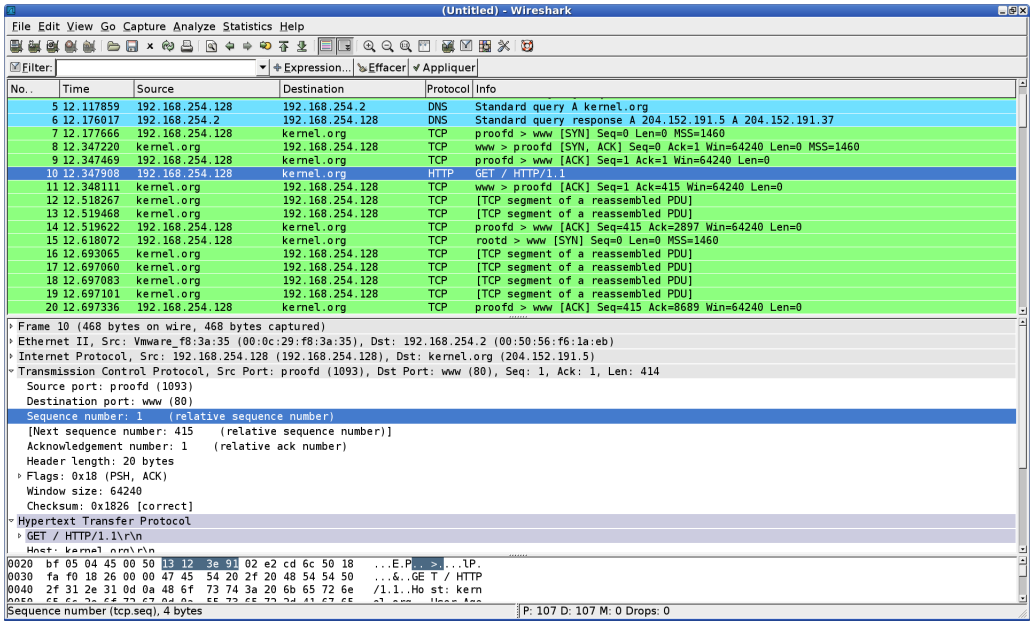
On modifie le contexte de travail du programme dumpcap.

```
# setcap cap_net_raw,cap_net_admin=eip /usr/bin/dumpcap
# getcap /usr/bin/dumpcap
/usr/bin/dumpcap = cap_net_admin,cap_net_raw+eip
```

- Les bits eip correspondent aux attributs **effective**, **inheritable** et **permitted**.
- Avec l'attribut **effective**, le noyau ne vérifie pas si l'UID vaut 0 (mode privilégié) si le programme nécessite une opération en mode privilégié.
- L'attribut **inheritable** transmet les aptitudes du processus actuel aux autres processus enfants.
- L'attribut **permitted** indique que le processus peut utiliser les aptitudes étendues du noyau Linux.

La documentation sur les **Linux Capabilities** est disponible à partir de la page Not needing root to administer Linux : lien 115.

3 Interface utilisateur



Capture d'écran Wireshark - vue complète

L'interface de l'analyseur se décompose en plusieurs barres ou fenêtres.

Barre de menus

On y retrouve la liste classique de menus. Voici une liste des fonctions remarquables accessibles à partir de ces menus.

- Le menu File sert à sauvegarder ou charger un fichier de capture réseau. Une capture peut très bien avoir été réalisée sur une sonde distante ou avec un autre outil et être analysée avec **Wireshark** à postériori.
- Le menu Capture sert à fixer les paramètres d'une nouvelle capture réseau. Voir Section 4, « Capture d'une série de trame » : lien 116.
- Le menu Statistics sert à effectuer différents calculs sur les volumes de données et la répartition des protocoles.

Barre des icônes

Cette barre regroupe tous les raccourcis sur les manipulations d'une capture.

Barre de filtrage

Cette barre sert à saisir l'expression de filtrage à postériori d'une capture pour isoler tout ou partie d'un échange réseau.

Fenêtre contenant la liste des trames capturées

Sur chaque ligne, on retrouve :

- le numéro du paquet ;
- son temps de capture ;
- sa source ;
- sa destination ;
- le protocole de plus haut niveau décodé ;
- le résumé des champs caractéristiques de ce protocole.

Fenêtre d'affichage de la pile des protocoles décodés pour la trame sélectionnée Avant toute opération de développement des champs d'un ou plusieurs protocoles, cette fenêtre donne la liste de la pile de protocoles décodés allant du niveau physique (en haut) jusqu'au niveau le plus haut reconnu (en bas). Le protocole de niveau le plus haut reconnu est celui qui apparaît dans la colonne protocole de la Fenêtre contenant la liste des trames capturées : lien 117.

- La première ligne ou niveau Frame correspond à une pseudocouche physique. Comme il n'est pas possible de réaliser la capture directement à partir des composants électroniques qui pilotent l'interface réseau sans perturber le fonctionnement du système, l'opération a lieu au niveau liaison à l'aide de la bibliothèque **libpcap**. À ce niveau, les informations disponibles sont : la quantité de bits capturés et la date de capture.
- La deuxième ligne correspond au niveau liaison. On y détaille le type, les champs de la trame et les adresses physiques.
- La troisième ligne correspond au niveau réseau. On y détaille les champs du protocole réseau reconnu : adresses logiques et indicateurs d'état.
- La quatrième ligne correspond au niveau transport. On y détaille les champs du protocole de transport reconnu : état de la connexion, numéros de ports utilisés et diverses options.

— La cinquième ligne correspond au niveau application. On y trouve les données utilisateur.

Pour le développement de chacun des champs de la trame, il faut cliquer sur le triangle situé à gauche au niveau de chaque couche.

Fenêtre d'affichage brut de la trame sélectionnée

Cette fenêtre affiche tous les octets de la trame en hexadécimal.

4 Capture d'une série de trames

Après avoir lancé le logiciel **Wireshark**, opérez la séquence suivante pour capturer une série de 60 trames :

1. Sélectionnez Capture puis Start ;
2. a ligne Capture Filter permet de préciser un filtrage **à priori**. La syntaxe de ce filtrage est identique à celle de la commande **tcpdump**. La documentation est disponible à partir des pages de manuels de cette commande : **man tcpdump**. Voici trois exemples :

   D'une façon plus générale, on peut combiner plusieurs critères avec les opérateurs logiques *and* et|ou *or*.

   En règle générale, il faut limiter au maximum le filtrage **à priori** de façon à disposer du maximum d'informations pour l'analyse. La syntaxe de la Section 5, « Filtrage de l'affichage après capture » (lien 118) offre beaucoup plus de possibilités.

   (a) **IP :** en spécifiant le protocole réseau à analyser, on évite la capture des trames des autres protocoles de niveau réseau (IPX) et des protocoles de niveau liaison (STP, CDP, etc.),

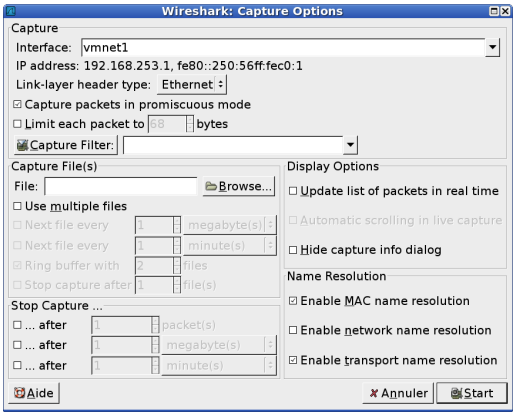
   (b) **Host 192.168.0.1 :** en spécifiant l'adresse IP d'un hôte, on ne retient que le trafic émis et reçu par cette adresse,

   (c) **Host 192.168.0.1 and host 10.0.0.1 :** en spécifiant les adresses IP de deux hôtes, on ne retient que le trafic entre ces deux adresses :

      i. Le type : host, net et port,

      ii. La direction : src et dst,

      iii. Le protocole : ether, fddi, tr, ip, ip6, arp, rarp, decnet, tcp et udp ;
3. La rubrique Stop Capture permet de fixer plusieurs critères d'arrêt en fonction du nombre de trames et|ou du volume de données capturées ;
4. Cliquez sur le bouton Valider pour lancer la capture.



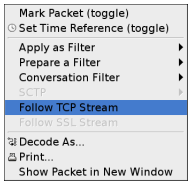
Paramètres de capture - vue complète

5 Filtrage de l'affichage après capture

Le filtrage **à postériori** est certainement l'étape la plus importante dans l'analyse réseau. C'est cette opération qui permet d'isoler l'information pertinente. La granularité de la syntaxe de filtrage disponible avec **Wireshark** est très importante. Il est possible de retenir un champ unique parmi les 820 protocoles supportés. Voici quelques exemples de filtrage allant du plus général au plus détaillé.

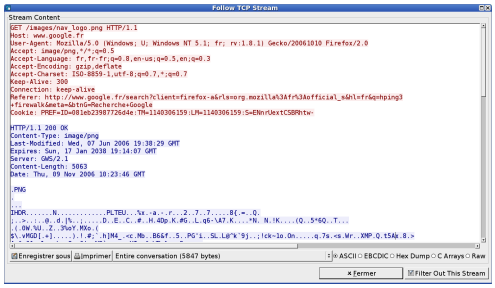
5.1 Isoler une connexion TCP

Après avoir réalisé une capture, il est possible d'isoler une connexion TCP en repérant son établissement (le début) et sa libération (la fin). En cliquant sur le bouton droit de la souris après avoir sélectionné n'importe quelle trame appartenant à la connexion à isoler, il faut valider l'option Follow TCP Stream.



Isoler une connexion TCP - vue complète

À la suite de cette opération, **Wireshark** ouvre une nouvelle fenêtre contenant les données vues de la couche transport.



Données vues de la couche transport - vue complète

5.2 Syntaxe du filtrage à postériori

Comme indiqué ci-avant, la granularité de la syntaxe de filtrage est très importante. Elle peut donc s'avérer très complexe à manipuler. **Wireshark** offre plusieurs solutions pour rendre l'apprentissage de cette syntaxe interactif.

Tout d'abord, l'opération précédente de filtrage simplifié (voir Section 5.1, « Isoler une connexion TCP » ci-dessus) n'était qu'un cas particulier de saisie interactive de filtre de capture. En sélectionnant l'option Follow TCP Stream, on a « saisi » un filtre avec la syntaxe suivante :

```
1 (ip.addr(1) eq 192.168.1.9(2) and ip.
      addr eq 80.247.225.35) \
2 and(3) (tcp.port(4) eq 32783(5) and tcp.
      port eq 80)
```

Cette expression est extraite de la barre de filtrage : lien 119. Elle doit tenir sur une ligne unique, quelle que soit sa longueur.

ip.addr : sélection d'une adresse IP ; **eq 192.168.1.9 :** valeur particulière d'adresse IP. L'opérateur eq correspond à un test d'égalité. Il est aussi possible d'utiliser la syntaxe du langage C pour les tests :

- == : égalité,
- != : différence,
- >= : supérieur ou égal,
- <= : inférieur ou égal ;

Les opérateurs logiques tels que *and* et|ou *or* associés aux parenthèses servent à composer des expressions de sélection précises ;

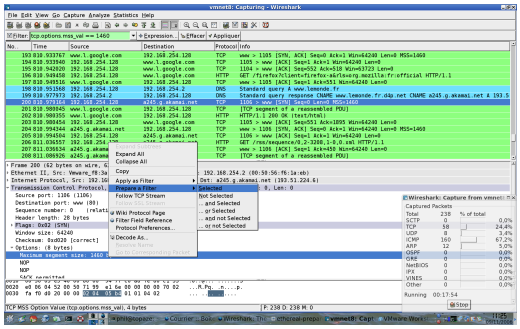
tcp.port : sélection d'un numéro de port du protocole TCP de la couche transport ;

eq 32783 : valeur particulière de port TCP. La syntaxe de test est identique pour tous les champs des différents protocoles reconnus.

La construction interactive des filtres d'affichage peut se faire à l'aide de la souris. Voici deux exemples «simplistes».

Option TCP MSS

Admettons que l'on veuille repérer toutes les trames capturées dans lesquelles l'option MSS (**Maximum Segment Size**) apparaît. On développe alors l'en-tête TCP d'un paquet correspondant à une demande de connexion pour faire apparaître cette option. En cliquant sur le bouton droit de la souris, on accède au menu Prepare a Filter.



Préparation d'un filtre d'affichage à la souris - vue complète

L'expression préparée apparaît dans le champ de la barre de filtrage (lien 120) :

```
1 tcp.options.mss_val == 1460
```

Supposons maintenant que l'on veuille afficher toutes les trames ayant cette option indépendamment de sa valeur. Il suffit alors de supprimer le test :

```
1 tcp.options.mss_val
```

Fragmentation IP

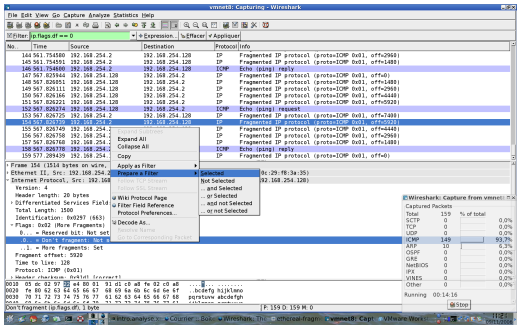
Admettons que l'on veuille observer la fragmentation IP en repérant les champs correspondants de l'en-tête des paquets IP. Tout d'abord, il faut «provoquer» la fragmentation IP artificiellement. On utilise deux hôtes avec chacun une interface Ethernet et un hub. En réduisant la taille maximum des données transmises par paquet (**Maximum Transmit Unit**) sur l'interface Ethernet d'un hôte, on observe plus facilement les effets de la fragmentation.

```
1
2   ----------
3  |_=_=_=_=_// Hub
4    |      |
5   /        \_______
6   |.....            \
7 .------,~            |.....
8 |Hote_A|'          .------,~
9 |       ||        |Hote_B|'
10\------ /         |       ||
11 ======/          \------ /
12192.168.254.128    ======/
13                  192.168.254.2
14
15Hote_A # ifconfig eth0 mtu 256
16Hote_A # ping -s 128 -c 5 192.168.254.2
17PING 192.168.254.2 (192.168.254.2) 128(1
       56) bytes of data.
18136 bytes from 192.168.254.2: icmp_seq=1
        ttl=64 time=0.591 ms
19136 bytes from 192.168.254.2: icmp_seq=2
        ttl=64 time=0.528 ms
20136 bytes from 192.168.254.2: icmp_seq=3
        ttl=64 time=0.554 ms
21136 bytes from 192.168.254.2: icmp_seq=4
        ttl=64 time=0.545 ms
22136 bytes from 192.168.254.2: icmp_seq=5
        ttl=64 time=0.546 ms
23
24--- 192.168.254.2 ping statistics ---
```

```
24 5 packets transmitted, 5 received, 0%
       packet loss, time 3999ms
25 rtt min/avg/max/mdev = 0.528/0.552/0.591
       /0.036 ms
26 Hote_A # ping -s 8192 -c 5 192.168.254.2
27 PING 192.168.254.2 (192.168.254.2) 8192(
       8220) bytes of data.
28 8200 bytes from 192.168.254.2: icmp_seq=
       1 ttl=64 time=15.4 ms
29 8200 bytes from 192.168.254.2: icmp_seq=
       2 ttl=64 time=15.4 ms
30 8200 bytes from 192.168.254.2: icmp_seq=
       3 ttl=64 time=15.4 ms
31 8200 bytes from 192.168.254.2: icmp_seq=
       4 ttl=64 time=15.4 ms
32 8200 bytes from 192.168.254.2: icmp_seq=
       5 ttl=64 time=15.4 ms
33
34 --- 192.168.254.2 ping statistics ---
35 5 packets transmitted, 5 received, 0%
       packet loss, time 4004ms
36 rtt min/avg/max/mdev = 15.444/15.462/15.
       481/0.079 ms
```

On observe ensuite le résultat sur l'affichage des trames capturées. La syntaxe du filtre est :

```
1 ip.flags.df == 0
```



Préparation d'un filtre d'affichage à la souris - vue complète

Connaissant maintenant la syntaxe d'identification de la fragmentation IP, il sera toujours possible d'appliquer le même filtre sur une capture beaucoup plus importante en volume.

5.3 Documentation de référence sur les filtres d'affichage

La documentation sur l'ensemble des champs des protocoles reconnus utilisables dans les expressions de filtres d'affichage est disponible à l'adresse : Display Filter Reference (lien 121).

6 Analyse à distance

Lorsque l'on exploite une infrastructure de serveurs avec plusieurs périmètres réseau cloisonnés, il est fréquent de devoir procéder à des captures réseau à distance. De plus, la plupart des serveurs récents sont des lames qui n'ont ni clavier ni écran. Voici donc un exemple de scénario capture réseau réalisée sur un hôte distant exploitée ensuite sur un poste de travail ayant une interface graphique.

Dans la série de copies d'écran ci-après, on considère les éléments suivants :

- le poste de travail sur lequel l'analyse est effectuée en mode graphique après collecte du fichier de capture est appelé <my_laptop.myothernet> ;
- le serveur lame sans écran ni clavier sur lequel la capture réseau est réalisée est appelé <my_distant_server.mynet>. On y accède via une console sécurisée SSH ;
- on suppose que les deux hôtes ont un compte utilisateur me. Le compte utilisateur sur le serveur doit disposer des droits nécessaires à la capture de trames sur les interfaces réseau du serveur. Ces droits sont gérés avec sudo ;
- on utilise l'application tshark qui permet d'exécuter l'analyse réseau directement à la console sans recours à une interface graphique. Cette application est fournie par le paquet **Debian** du même nom. Voir le résultat de la commande **$ apt-cache show tshark** pour obtenir les informations sur ce paquet ;
- les indications données ci-dessous ne peuvent se substituer aux pages du manuel de l'application. Il est vivement conseillé de les consulter pour adapter l'analyse réseau à ses besoins : **man tshark**.

Connexion au serveur depuis le poste de travail

Comme indiqué ci-avant, on accède au serveur via une console sécurisée SSH. À partir de Windoze, l'outil putty permet d'effectuer la même opération.

```
1  me@<my_laptop>:~$ ssh me@<
       my_distant_server.mynet>
2
3  Linux <my_distant_server> 2.6.15 #1 SMP
       Mon Mar 13 14:54:19 CET 2006 i686
4
5  The programs included with the Debian
       GNU/Linux system are free software;
6  the exact distribution terms for each
       program are described in the
7  individual files in /usr/share/doc/*/
       copyright.
8
9  Debian GNU/Linux comes with ABSOLUTELY
       NO WARRANTY, to the extent
10 permitted by applicable law.
11 No mail.
12 Last login: Tue Mar 21 10:45:38 2006
       from <my_laptop.myothernet>
13
14 me@<my_distant_server>:~$
```

Lancement de la capture réseau dans un nouveau shell

Un utilisateur «normal» n'ayant pas les droits suffisants pour accéder directement aux interfaces réseau, on doit lancer l'analyseur de réseau via sudo :
$ sudo tshark

On lance cette commande dans un nouveau shell en ajoutant le symbole & à la fin de la ligne. De cette façon, on conserve la possibilité de lancer d'autres commandes sur la console obtenue lors de la connexion au serveur.

```
1  me@<my_distant_server>:~$ sudo tshark -q
       -i _eth0 -w distant.cap \
2    -a filesize:4096 tcp and ! host <
       my_laptop.myothernet> &
```

— L'option -q rend la capture «silencieuse». Il s'agit surtout de supprimer l'affichage du compte des paquets enregistrés pendant la capture. Cet affichage est gênant si l'on souhaite conserver la console pour effectuer d'autres manipulations en cours de capture.

— L'option -i _eth0 désigne l'interface réseau sur laquelle la capture est réalisée.

— L'option -w distant.cap désigne le fichier dans lequel les paquets capturés sont enregistrés. Sans spécification du format de fichier avec l'option -F, les paquets capturés sont enregistrés directement (mode **raw**).

— L'option -a filesize :4096 donne le critère d'arrêt de l'enregistrement. Ici, le critère retenu est la taille du fichier de capture. Cette taille est comptabilisée en multiple du kilooctet (1 024 octets) ; soit 4 096 ko dans cet exemple.

— Les options suivantes correspondent au filtrage à priori des paquets à enregistrer. On spécifie le protocole de transport TCP **et** on n'enregistre pas les paquets de l'hôte qui a ouvert la console sécurisée :! host <my_laptop.myothernet>. Sans cette dernière précaution, l'enregistrement ne contiendra pratiquement que les échanges SSH. Ces échanges sont sans intérêt puisqu'ils correspondent aux communications entre les deux hôtes utilisés pour l'analyse distante.

«Initiation» du trafic réseau à capturer.

Cette commande n'est qu'un prétexte pour remplir le fichier de capture. Avec le téléchargement d'une image des sources du noyau Linux, on est sûr de faire transiter un volume suffisant ;-).

```
1  me@<my_distant_server>:~$ wget \
2    http://kernel.org/pub/linux/kernel/v2.
       6/linux-2.6.16.tar.bz2
3  --11:14:29--  http://kernel.org/pub/
     linux/kernel/v2.6/linux-2.6.16.tar.
     bz2
4           => 'linux-2.6.16.tar.bz2'
5  Résolution de kernel.org... 204.152.191.
     5, 204.152.191.37
6  Connexion vers kernel.org|204.152.191.5
     |:80...connecté.
7  requête HTTP transmise, en attente de la
      réponse...200 OK
8  Longueur: 40 845 005 (39M) [application/
     x-bzip2]
9
10 100%[===============//=================
     ===>] 40 845 005   296.19K/s    ETA
     00:00
11
12 11:16:58 (292.09 KB/s) - \of linux-2.6.1
     6.tar.bz2 \fg sauvegardé [40845005/4
     0845005]
```

Fin de la capture et visualisation du fichier

Comme indiqué ci-avant, l'enregistrement s'arrête lorsque le fichier atteint la taille de 4 096 ko.

```
1  [1]+  Done    sudo tshark -q -i _eth0 -w
     distant.cap \
2                    -a filesize:4096 tcp
                       and ! host <
                       my_laptop.
                       myothernet>
3
4  me@<my_distant_server>:~$ ls -lAh
5  -rw-------  1 root latu   4,1M 2006-03-2
     1 11:14 distant.cap
6  -rw-r--r--  1 latu latu    39M 2006-03-2
     0 07:22 linux-2.6.16.tar.bz2
7
8  me@<my_distant_server>:~$ sudo chmod 640
      distant.cap
9
10 me@<my_distant_server>:~$ exit
11 logout
12 Connection to <my_distant_server.mynet>
     closed.
```

L'enregistrement sur fichier ayant été réalisé avec l'identité du super-utilisateur via la commande **sudo**, il faut changer le masque des permissions de ce fichier ou son propriétaire. Dans cet exemple, c'est le masque des permissions d'accès qui a été étendu pour que l'utilisateur normal puisse lire le fichier de capture et le transférer sur son poste de travail.

Récupération du fichier de capture sur le poste de travail

```
1  me@<my_laptop>:~$ scp me@<
     my_distant_server.mynet>:~/distant.
     cap .
2  distant.cap

     100% 4097KB 682.8KB/s   00:06
3
4  me@<my_laptop>:~$ wireshark -r distant.
     cap
```

La commande **scp** illustre le transfert du fichier de capture réseau via SSH. On peut effectuer la même opération à partir de Windoze avec l'outil WinSCP.

Enfin, il est possible de lire le fichier de capture directement au lancement de l'analyseur réseau avec l'option -r.

7 Travaux pratiques : navigation Web (HTTP)

7.1 Protocoles étudiés

— Adressage matériel (MAC|Ethernet) et logique (IP).

— Requête et réponse du service de noms de domaine (DNS).

— Établissement, maintien et libération de connexion TCP : procédure en trois étapes, numéros de séquence et d'acquittement.

— Requête et réponse HTTP.

7.2 Marche à suivre

1. Lancez **Wireshark**.
2. Lancez la capture des trames sans restrictions d'adresses, de protocoles ou de volume.
3. Lancez un navigateur Web et saisissez une adresse de site (URL) de votre choix.
4. Une fois la page complètement chargée, arrêtez la capture. Sauvegardez un fichier de capture.
5. Passez aux questions suivantes.

Suivant le contexte de connexion, le volume d'informations capturé varie énormément : connexion DSL, réseau local commuté ou non, multiplicité des protocoles réseau, etc. Il est cependant préférable d'effectuer la première capture sans aucune restriction **à priori** de façon à avoir une image exacte du trafic. Si l'information utile est vraiment noyée dans du «bruit», il est toujours possible de reprendre la capture avec un filtre ; voir Section 4, « Capture d'une série de trames ».

7.3 Analyse des protocoles

Pour répondre aux questions suivantes, utilisez le résultat de la capture issue de l'étape précédente ou chargez un fichier de capture.

7.3.a Protocoles capturés

Q1.	Quels sont les protocoles indiqués dans la colonne « Protocol » de la fenêtre de liste des trames capturées ? Confirmez que la capture contient bien les protocoles DNS, TCP et HTTP.

7.3.b Trame Ethernet, paquet IP et datagramme UDP

Analysez la trame correspondant au premier message DNS émis par le client Web.

Q2.	Quelles sont les adresses (MAC|Ethernet) et IP du client ?
Q3.	Quel est le contenu du champ type de la trame Ethernet ?
Q4.	Quelles sont les adresses destination (MAC|Ethernet) et IP ?
Q5.	À quelles machines correspondent ces adresses ?

Analysez l'en-tête IP du premier message DNS émis par le client Web.

Q6.	Quelle est la taille de l'en-tête ? Quelle est la longueur totale du paquet ?
Q7.	Repérez le champ « type de protocole » dans l'en-tête. Quel est le numéro et le type de protocole présents dans les données du paquet ?

Analysez l'en-tête UDP du premier message DNS émis par le client Web.

Q8.	Quels sont les numéros de ports du client et du serveur ? Quelles sont les particularités de ces valeurs ? Quel est le protocole de couche application présent dans les données du message ?
Q9.	Quelle est la valeur indiquée dans le champ longueur de l'en-tête UDP ? Est-ce qu'elle correspond à l'information donnée dans l'en-tête du paquet IP ?

Faites un croquis des piles de protocoles des couches physiques à application pour le client et le serveur ; identifiez les unités de données de protocoles (PDU) et les communications de bout en bout.

7.3.c Service DNS

Analysez le message de requête DNS émis par le client Web.

Q10.	Quel est le champ qui indique si le message est une requête ou une réponse ?
Q11.	Quelle est l'information transportée dans le corps de la requête ? Identifier le type et la classe de la requête.
Q12.	Quel est l'identificateur de transaction de la requête ?

On considère maintenant la réponse à la requête précédente.

Q13.	Quelles devraient être les adresses (MAC|Ethernet) et IP de ce paquet ? Vérifiez que les adresses attendues sont présentes.
Q14.	Quelle est la taille du paquet IP ; du message UDP ? Cette taille est-elle plus importante que celle du paquet de la requête
Q15.	Quel est l'identificateur de transaction de la réponse ? Est-ce qu'il correspond à la requête ?
Q16.	Combien de réponses sont disponibles dans le message de réponse ? Comparez les réponses et leurs valeurs TTL (**Time-to-live**).

7.3.d Connexion TCP

Identifiez la trame qui correspond au premier segment TCP dans la procédure en trois étapes (**three ways handshake**) qui initie la connexion entre le client et le serveur HTTP.

Q17.	Quelles sont les adresses (MAC|Ethernet) et IP attendues pour cette trame ? Quelles sont les valeurs des champs « type » et « protocole » respectivement attendus pour cette trame et ce paquet ? Vérifiez que ces champs et adresses correspondent.
Q18.	Expliquez les valeurs des adresses destination (MAC|Ethernet) et IP ? À quels hôtes correspondent ces adresses ?
Q19.	Identifiez les numéros de ports utilisés par le client. Pourquoi ces valeurs sont-elles utilisées ?
Q20.	Quelle est la longueur du segment TCP ?
Q21.	Quel est le numéro de séquence initiale (**Initial Sequence Number** ou ISN émis par le client vers le serveur ? Quelle est la taille de fenêtre initiale ? Quelle est la taille maximale de segment (**Maximum Segment Size** ou MSS) ?
Q22.	Trouvez la valeur hexadécimale de l'octet qui contient l'indicateur d'état SYN ?

Identifiez la trame qui correspond au second segment TCP dans la procédure en trois étapes (**three ways handshake**).

Q23.	Combien de temps s'est écoulé entre la capture du premier et du second segment TCP ?
Q24.	Relevez les valeurs des champs suivants de cette trame : — adresses MAC source et destination de la trame Ethernet ; — adresses source et destination du paquet IP ; — numéros de séquence et d'acquittement du segment TCP ; — valeurs des indicateurs d'état. Vérifiez que tout correspond aux valeurs attendues.
Q25.	Quelle est la longueur du segment TCP ?
Q26.	Quel est le numéro de séquence initial (**Initial Sequence Number** ou ISN émis par le serveur vers le client ? Quelle est la taille de fenêtre initiale ? Quelle est la taille maximale de segment (**Maximum Segment Size** ou MSS) ?

Identifiez la trame qui correspond au dernier segment TCP dans la procédure en trois étapes (**three ways handshake**).

Q27.	Combien de temps s'est écoulé entre la capture du second et du troisième segment TCP ? Comparez cette valeur avec celle relevée entre le premier et le second segment et expliquez la différence.
Q28.	Relevez les valeurs des champs suivants de cette trame : — numéros de séquence et d'acquittement du segment TCP ; — valeurs des indicateurs d'état ; — tailles de fenêtre ; Vérifiez que tout correspond aux valeurs attendues.
Q29.	Quelle est la longueur du segment TCP ?

7.3.e Requête HTTP GET

Identifiez la trame qui correspond au message HTTP GET.

Q30.	Quelles sont les valeurs des numéros de séquence et d'acquittement de l'en-tête TCP ? Vérifiez que tout correspond aux valeurs attendues.
Q31.	Quels sont les indicateurs d'état actifs de l'en-tête TCP ? Expliquez pourquoi.
Q32.	Quelles sont les longueurs de l'en-tête et de la «charge» du message TCP ?

On considère maintenant le contenu du message HTTP GET.

Q33.	Comparez le texte décodé dans la fenêtre d'affichage de la pile de protocoles (lien 122) avec le contenu de la fenêtre d'affichage brut (lien 123).
Q34.	Comptez le nombre d'octets du message et vérifier que ce nombre correspond au champ longueur de l'en-tête TCP.
Q35.	Quel est le prochain numéro de séquence attendu dans le message suivant émis par le serveur HTTP ?

7.3.f Réponse HTTP

Q36.	Combien de temps s'est écoulé entre la capture du message GET et la capture du message de réponse correspondant ?
Q37.	Déterminez si le serveur répond avec un message HTTP ou un segment TCP ACK ?
Q38.	Quel est le numéro de séquence émis par le serveur HTTP ? Est-ce qu'il correspond à la valeur attendue ?

On considère maintenant l'en-tête du message réponse HTTP.

Q39.	Quelle est la longueur de la «charge» indiquée dans l'en-tête TCP ?
Q40.	Quels sont les indicateurs d'état actifs de l'en-tête TCP ? Expliquez pourquoi.
Q41.	Quel est le prochain numéro de séquence attendu dans le message suivant émis par le client ?

On considère maintenant le corps du message réponse HTTP.

Q42.	Quel est le code dans le message de réponse ?
Q43.	Sélectionnez ce code avec la souris dans la fenêtre d'affichage de la pile de protocoles (lien 124) et comparez-le avec ce qui est affiché sur la page du navigateur Web. Cette opération revient à suivre la démarche présentée dans la Section 5.1, « Isoler une connexion TCP ».

8 Travaux pratiques : messages de contrôle Internet (ICMP)

8.1 Protocoles et outils étudiés

- **Internet Control Message Protocol** ou ICMP ; messages de type : Echo, Echo Reply et Time Exceeded.
- **Internet Protocol** ou IP ; champ de l'en-tête IP : Time to Live.
- Commande **ping**.
- Commandes **traceroute** et **tcptraceroute**.

8.2 Marche à suivre

Commande ping

1. Lancez **Wireshark**.
2. Lancez la capture des trames sans restrictions d'adresses, de protocoles ou de volume.
3. Lancez une console et tapez une commande du type **ping -c10 www.phrack.org**. L'option -c10 limite le nombre de requêtes ICMP à 10. Bien sûr, le choix de l'adresse à contacter est totalement libre.
4. Arrêtez la capture lorsque l'invite de commande réapparaît à la console.
5. Sauvegardez le fichier de capture.

Commande traceroute

1. Lancez **Wireshark**.
2. Lancez la capture des trames sans restrictions d'adresses, de protocoles ou de volume.
3. Lancez une console et tapez une commande du type **traceroute www.phrack.org**. Bien sûr, le choix de l'adresse à contacter est totalement libre.
4. Arrêtez la capture lorsque l'invite de commande réapparaît à la console.
5. Sauvegardez le fichier de capture.

La plage de ports UDP utilisée par défaut par la commande **traceroute** est de plus en plus fréquemment bloquée par les équipements d'interconnexion. Il est alors utile d'envisager l'emploi de la commande **tcptraceroute** avec laquelle on peut fixer les ports source et destination.

Commande tcptraceroute

1. Lancez **Wireshark**.
2. Lancez la capture des trames sans restrictions d'adresses, de protocoles ou de volume.
3. Lancez une console et tapez une commande du type **tcptraceroute -p 1024 www.phrack.org 80**. Bien sûr, le choix de l'adresse à contacter est totalement libre.
4. Arrêtez la capture lorsque l'invite de commande réapparaît à la console.
5. Sauvegardez le fichier de capture.

8.3 Analyse avec ping

Pour répondre aux questions suivantes, utilisez le résultat de la capture issue de l'étape précédente ou chargez un fichier de capture.

8.3.a Protocoles capturés

Q44.	Quels sont les protocoles indiqués dans la colonne Protocol de la fenêtre de liste des trames capturées ? Il est probable que les paquets ICMP soient précédés d'un jeu de questions/réponses DNS.
Q45.	Relevez l'adresse IP renvoyée avec la réponse DNS.

8.3.b Message ICMP «Echo Request»

Étude du paquet IP qui correspond au premier message ICMP Echo Request.

Q46.	Quelle est l'adresse IP destination du paquet ? Quelle est la valeur du champ Protocol Type ? Quelle est la valeur du champ Time to Live ?

Étude du message ICMP.

Q47.	Quel est le type de message ICMP ? Quel est l'identificateur de message ? Quel est le numéro de séquence ?
Q48.	Sélectionnez à la souris les octets de données du message de requête. Comparez ces données avec celles affichées dans la fenêtre d'affichage brut : lien 125.

8.3.c Message ICMP «Echo Reply»

Étude du paquet IP qui correspond au premier message ICMP Echo Reply.

Q49.	Quelles sont les adresses IP source et destination du paquet ? Quelle est la valeur du champ Protocol Type ? Quelle est la valeur du champ Time to Live ?

Étude du message ICMP.

Q50.	Quel est le type de message ICMP ? Comparez l'identificateur de message et le numéro de séquence du message de réponse avec les valeurs du message de requête.
Q51.	Sélectionnez à la souris les octets de données du message de requête. Comparez ces données avec celles affichées dans le message de requête.

8.3.d Messages ICMP restants

Reprenez les deux points précédents pour les messages ICMP Echo Request et Echo Reply restants.

Q52.	Comment les champs d'identification et de numéro de séquence évoluent-ils dans le temps ?
Q53.	Est-ce que les séquences de données des requêtes et des réponses changent ?
Q54.	Calculez l'écart de temps entre l'émission de chaque message Echo Request et la réception de chaque message Echo Reply. Comparez les résultats avec les valeurs maximales, moyennes et minimales fournies par la commande **ping**.

8.4 Analyse avec (tcp)traceroute

Pour répondre aux questions suivantes, utilisez le résultat de la capture issue de l'étape précédente ou chargez un fichier de capture.

8.4.a Protocoles capturés

Q55.	Quels sont les protocoles indiqués dans la colonne Protocol de la fenêtre de liste des trames capturées ? Il est probable que les paquets ICMP soient précédés d'un jeu de questions/réponses DNS.
Q56.	Relevez l'adresse IP renvoyée avec la réponse DNS.

8.4.b Message UDP

Q57.	Quelle est l'adresse IP destination du premier paquet contenant le message UDP ? Quelles sont les valeurs des champs Protocol Type et Time to Live ? Comparez l'adresse IP destination relevée avec celle de la réponse DNS. Notez les valeurs caractéristiques de l'en-tête IP en vue d'une utilisation ultérieure : lien 126.
Q58.	Combien d'octets de données sont présents dans ce message de requête ? Notez la séquence de caractères présents dans la troisième fenêtre.

8.4.c Message ICMP «Time Exceeded»

Q59.	Quelles sont les adresses IP source et destination du paquet de la première réponse ICMP Time Exceeded ?

Étude du message ICMP.

Q60.	Quel est le type de message ICMP ? Les champs Type, Code et Checksum sont suivis par plusieurs octets à zéro, puis par l'en-tête IP du message ICMP Echo Request. Comparez les valeurs caractéristiques de cet en-tête avec celles notées ci-avant : lien 127.
Q61.	Est-ce que le message ICMP contient de nouveaux octets de données ?

8.4.d Évolution du champ TTL

Q62.	Combien de messages UDP sont émis avec la même valeur de champ TTL dans l'en-tête de paquet IP ?
Q63.	Quelles sont les adresses IP source des paquets ICMP Time Exceeded ? Comparez ces adresses avec celles données lors de l'exécution de la commande **traceroute**.
Q64.	Quel est le type du message ICMP reçu lorsque l'hôte destinataire est atteint ?
Q65.	Comment calculer les temps affichés par la commande **traceroute** à partir des valeurs données dans la colonne Time de la fenêtre des trames capturées ? Utilisez les pages de manuels de la commande **traceroute** pour obtenir la signification des différentes valeurs de temps pour atteindre une destination.

8.4.e Variantes

Il est possible de reprendre les questions ci-dessus en utilisant différentes options des commandes **traceroute** et|ou **tcptraceroute**.

- Analyse uniquement à base de messages ICMP avec l'option -I : **traceroute -I www.phrack.org**.
- Analyse à base de segments TCP en précisant le numéro de port visé : **tcptraceroute www.phrack.org 80**. Cette dernière variante est très utile pour vérifier si un service est ouvert ou non.

9 Documents de référence

Guide de l'utilisateur

- Le Wireshark User's Guide est la référence la plus complète sur l'utilisation de notre analyseur de trafic favori : lien 128 !

Protocoles

- Le fichier PDF TCP/IP and tcpdump Pocket Reference Guide est une «antisèche» sur les champs des en-têtes des protocoles essentiels ; un document **indispensable** pour la pratique de l'analyse réseau : lien 129.

Travaux pratiques

- Le support Configuration d'une interface de réseau local présente les opérations de configuration d'une interface réseau et propose une exploitation des protocoles TCP/IP et ICMP sans recours à un analyseur réseau : lien 130.
- Le chapitre Using Ethereal - Chapter 4 of Ethereal packet sniffing est un extrait de livre consacré à la version antérieure de l'analyseur de trafic réseau : lien 131.
- Le site Ethereal Labs présente d'autres travaux pratiques basés sur **Ethereal**, la version antérieure de l'analyseur de trafic réseau : lien 132.

*Retrouvez l'article de **Philippe Latu** en ligne : lien 133*

Les générateurs de nombres aléatoires

Cet article est une présentation de diverses méthodes pour générer des nombres pseudo-aléatoires.

1 Introduction

Comme dans toute activité de développement logiciel, le développement des jeux vidéo nécessite des tests et des corrections (et même beaucoup) ce qui implique souvent la nécessité de pouvoir reproduire un certain état du jeu. En cas de bogue dans un système de combat par exemple, on peut avoir besoin de reproduire à volonté et à l'identique un certain comportement des adversaires du joueur pour localiser le problème et ensuite le corriger.

La spécificité des jeux vidéo par rapport à d'autres applications, dans le domaine de la gestion par exemple, est que ce sont des applications qui font très souvent appel, et parfois très massivement, à la génération de nombres aléatoires. Mais ce caractère intrinsèquement aléatoire de bon nombre de jeux entre souvent en conflit avec les besoins de reproductibilité.

Dans la plupart des langages, il existe une fonction ou méthode permettant de générer des nombres aléatoires à partir de ce qui est communément appelé graine ou *seed* en anglais, et qui permet de reproduire à l'identique une séquence de nombres aléatoires.

Mais ce n'est pas le cas de tous les langages. Par exemple, en JavaScript, la fonction de base, *Math.random()*, ne donne aucun moyen au développeur de définir une graine et n'a donc aucun moyen de reproduire une séquence aléatoire générée plus tôt. C'est un peu comme au cinéma où le spectateur n'a aucune possibilité de pause ou de retour arrière. Même si cela fait partie de l'expérience du cinéma, dans le domaine du développement c'est autrement plus gênant.

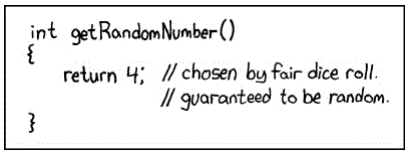
Le but de cet article sera de présenter différents générateurs de nombres aléatoires (ou RNG pour *Random Number Generator* en anglais) se basant sur une graine pour générer une séquence de nombres aléatoires.

Cet article peut s'adresser aux développeurs utilisant d'autres langages que JavaScript. Soit parce que leur langage ne fournit pas non plus un générateur de séries aléatoires reproductibles, soit parce que la qualité du générateur par défaut n'est pas suffisante pour les besoins.

Le langage utilisé pour illustrer les algorithmes sera **TypeScript** qui est un sur-ensemble de JavaScript avec typage et héritage. Une présentation de ce langage est disponible ici : lien 134.

Aussi, une **démonstration** en ligne de cet article est disponible ici : lien 135.

2 Génération pseudo-aléatoire



Bien que le hasard soit pratiquement omniprésent dans la vie de tous les jours, si on admet que les lois de la physique quantique sont vraies, il est loin d'être trivial, encore à l'heure actuelle de produire des nombres véritablement aléatoires à l'aide d'un ordinateur à cause de son mode de fonctionnement purement déterministe (à condition de faire abstraction des bogues et des plantages). Sauf à le relier à un processus physique comme la désintégration d'un élément radioactif (lien 136) ou un bruit de fond, les ordinateurs actuels doivent recourir à des algorithmes déterministes pour simuler le hasard, ce qui est quelque part paradoxal.

Mais de l'ordre peut naître le chaos. Un exemple basique est de considérer le cas du nombre PI qui est un nombre parfaitement déterministe mais dont le développement décimal est dans une certaine mesure imprévisible, en tout cas revêt beaucoup des caractéristiques du hasard. (cf. Preuss : lien 137)

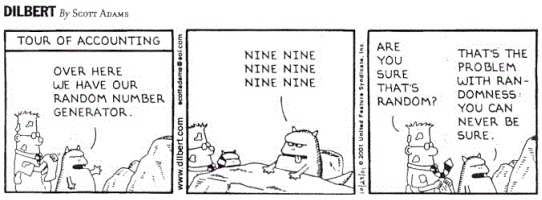
Dans le domaine qui nous intéresse, les jeux vidéo, qui s'apparente pour beaucoup à celui de la simulation numérique, il faut être capable de produire beaucoup de nombres aléatoires, et rapidement, sans avoir nécessairement la chance de jouer sur un supercalculateur. Pour cela, les mathématiciens ont trouvé au fil du temps diverses méthodes pour produire des nombres qui ont les apparences du hasard, même s'il faut toujours garder à l'esprit que ce n'est qu'une illusion.

Parmi les générateurs de nombres aléatoires, celui qui s'est imposé avec l'avènement des ordinateurs est le **générateur congruentiel linéaire**, et il est encore présent dans d'innombrables applications encore aujourd'hui.

À noter qu'en toute rigueur, les nombres géné-

rés par une procédure algorithmique déterministe comme ce sera le cas dans ce document, sont appelés des nombres pseudo-aléatoires. Cependant pour simplifier le propos, ils seront simplement appelés nombres aléatoires tout comme nous appellerons générateurs aléatoires, des procédures *stricto sensu* pseudo-aléatoires.

3 Générateur congruentiel linéaire (LCG)



3.1 Théorie

Un Générateur Congruentiel Linéaire (lien 138) ou LCG pour son acronyme en anglais est essentiellement une suite mathématique relativement simple, définie de la manière suivante :

— $X_{n+1} = (a * X_n + c)$ **mod** m

où **mod** désigne l'opérateur modulo, le reste de la division entière.

Le premier élément de la suite, X_0 est souvent nommé *seed* dans le jargon informatique. C'est la graine qui engendrera tous les autres nombres pseudo-aléatoires.

Les coefficients $<a, c, m>$ (resp. multiplicateur, incrément, modulo) définissent complètement un LCG et doivent obéir à certaines contraintes pour que le LCG produise des nombres aléatoires de qualité satisfaisante, ce qui n'est pas toujours le cas comme nous le verrons un peu plus bas.

C'est sur la base de ce modèle simple voire simpliste qu'encore aujourd'hui une majorité de nombres aléatoires sont produits sur nos ordinateurs. Cette famille de générateurs de nombres aléatoires a des défauts que nous allons voir, mais a trois principales qualités :

1. Les LCG sont très simples à implémenter.
2. Les LCG s'appuient sur un processus reproductible via la graine.
3. Les LCG sont rapides par rapport à d'autres générateurs de nombres aléatoires.

Ces trois qualités suffisent à expliquer la présence encore très importante des LCG dans les implémentations de langages.

Passons donc en revue les différents LCG.

3.2 RANDU

C'est le nom d'un algorithme mis au point chez IBM dans les années 1960. L'algorithme RANDU (lien 139) fait partie d'une classe particulière de LCG nommée Park-Miller d'où dérive un autre générateur bien connu, le MINSTD. Les LCG de la classe Park-Miller ont la particularité d'avoir un incrément c qui vaut zéro. Les générateurs de ce type sont donc définis par seulement deux nombres $<a, m>$ au lieu de trois avec la formule suivante :

— $X_{n+1} = (a * X_n)$ **mod** m

Les contraintes sont plus fortes que dans le cas général des LCG, avec par exemple le modulo m qui doit être un nombre premier et aussi une contrainte sur la graine initiale X_0 qui doit être impaire, et idéalement un nombre premier.

L'algorithme RANDU est défini par le couple $<65539, 2^{31}>$ soit la formule de récurrence :

— $X_{n+1} = (65539 * X_n)$ **mod** 2^{31}

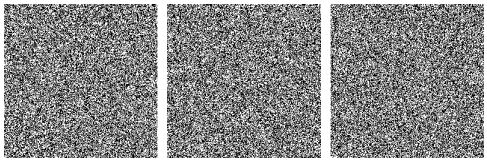
On constate que le modulo n'est pas un nombre impair et encore moins un nombre premier, même s'il n'est pas loin de 2^{31}-1, ce qui posera problème si on analyse un minimum la séquence des nombres aléatoires générés par un tel algorithme.

Son implémentation en TypeScript pourrait être la suivante :

```
1  var randomSeed = Date.now();
2
3  function randU(): number {
4    randomSeed = (randomSeed * 65539) % 21
         47483648;
5    return randomSeed / 2147483648.0;
6  } // randU
```

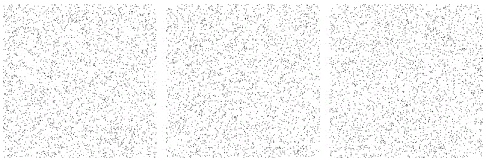
où *randomSeed* est une variable globale initialisée au préalable d'un appel à la fonction. Même s'il est une pratique courante d'initialiser la graine avec l'heure courante, en toute rigueur il faudrait l'initialiser avec une valeur plus chaotique (ayant plus d'entropie) comme le mouvement de la souris.

Il existe tout un tas de tests pour s'assurer de la qualité des générateurs de nombres aléatoires. Nous nous limiterons dans cet article à une simple analyse visuelle. Une série de nombres aléatoires binaires peut être représentée sous la forme d'une suite linéaire de pixels noirs ou blancs avec retour à la ligne. C'est ce qu'on pourrait appeler la représentation *pile ou face*. Un générateur aléatoire de qualité doit donner une image sans motif particulier. Un bon brouillard uniforme.



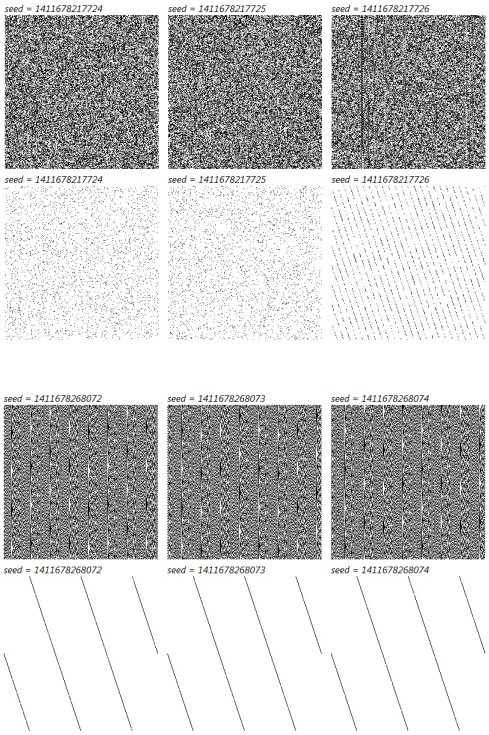
Représentations pile ou face typiquement aléatoires

Une autre façon simple de représenter une série de nombres aléatoires est une représentation dans un plan où deux nombres aléatoires consécutifs fournissent l'abscisse et l'ordonnée d'un point. Cela donne un nuage de points qui permet de voir certaines régularités. Évidemment, un générateur parfaitement aléatoire doit engendrer un nuage de points sans motif particulier et bien dispersé.



Nuages de points typiquement aléatoires

Dans le cas de RANDU, voici ce que nous obtenons :



Représentations pile ou face et nuages de points de RANDU

On peut constater que pour certaines valeurs de graines, la série de nombres engendrée par RANDU comporte des régularités flagrantes ce qui explique pourquoi cet algorithme est aujourd'hui tombé en désuétude.

L'algorithme RANDU est en fait célèbre, non pas pour son efficacité, très médiocre, voire mauvaise, comme on peut le constater, mais justement pour être un bon exemple de ce que ne doit pas être un LCG.

Depuis heureusement, on a fait mieux, quoique...

3.3 Windows

Après IBM, on va s'attaquer à un autre mastodonte de l'informatique : Microsoft. Car pour ceux qui ont utilisé les API C/C++ de Windows, il existe une fonction *rand()* par défaut, fournie dans les API du célèbre système d'exploitation.

Il se trouve que le générateur aléatoire par défaut de Windows est aussi un LCG, du moins jusqu'à la version XP.

Ses paramètres sont <214013, 2531011, 216> ce qui donne la suite :

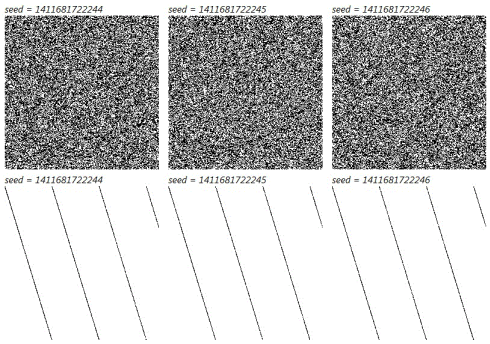
— $X_{n+1} = (214013 * X_n + 2531011)$ **mod** 2^{16}

En TypeScript l'implémentation pourrait être celle-ci :

```
1  var randomSeed = Date.now();
2
3  function randMSWin(): number {
4    randomSeed = randomSeed * 214013 + 253
         1011;
5    randomSeed = (randomSeed / 65536) % 32
         768; // extract bits 30..16
6    return randomSeed / 32767.0;
7  } // randMSWin
```

On remarque une petite variante qui consiste à n'extraire qu'une partie du résultat. En effet, il a été démontré que dans le cas des LCG, les bits de poids faibles avaient tendance à ne pas être très aléatoires, ce qui explique pourquoi dans beaucoup d'implémentations informatiques de LCG, une opération d'extraction des bits de poids forts est réalisée.

Si maintenant on visualise ce que nous donne ce générateur, on obtient ceci :



Représentations pile ou face et nuages de points du générateur de Windows

Si on ne considère que la représentation *pile ou face*, le générateur de Windows semble tout à fait valable dans la mesure où il est impossible de distinguer des régularités à l'œil nu.

Cependant, si on considère la représentation en nuage de points, c'est une autre affaire. On constate que la plupart des points sont alignés le long de quelques droites et ce de façon systématique. Donc même si c'est moins pathologique que pour RANDU, le générateur aléatoire de Windows est à éviter dans la mesure du possible.

En remarque, quand on voit ce que des firmes telles qu'IBM et Microsoft avec les moyens qui vont avec, aient pu produire des générateurs d'aussi piètre qualité, cela devrait rassurer les modestes développeurs que nous sommes.

3.4 Central Randomizer de Hoole

À une époque maintenant reculée, où il fallait se connecter à Internet via un modem bruyant et que le HTML n'était que vaguement implémenté dans les quelques navigateurs de l'époque, il n'existait pas de fonction native en JavaScript pour générer un nombre aléatoire. C'est à cette époque qu'un jeune homme, Paul Hoole de son état, a mis au point un petit LCG capable de combler cette lacune.

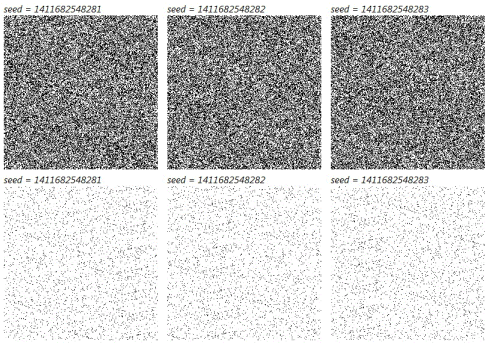
Ce qui a fini par s'appeler le Central Randomizer (lien 140) de Hoole est un LCG définit par le triplet <9301, 49297, 233280> et dont la suite caractéristique est :

— $X_{n+1} = (9301 * X_n + 49297)$ **mod** 233280

```
1  var randomSeed = Date.now();
2
3  function randCentral(): number {
4    randomSeed = (randomSeed * 9301 + 49297) % 233280;
5    return randomSeed / 233280.0;
6  } //randCentral
```

La principale différence par rapport à ses prédécesseurs est que le multiplicateur utilisé est petit par rapport à ceux utilisés plus haut. Cela tend à limiter les débordements de calcul lors de la multiplication avec la graine qui est souvent un grand nombre et permet dans l'absolu une meilleure rapidité bien que cela fût surtout important à l'époque des premiers navigateurs.

Une rapide analyse visuelle nous montre que les nombres aléatoires générés par ce générateur ne sont pas mauvais, même s'il est possible de relever de légères régularités.



Représentations pile ou face et nuages de points du générateur de Hoole

Dans un environnement optimal, on préférera toujours un autre générateur que celui-ci. Cependant, dans le cas d'une application JavaScript nécessitant un générateur de nombres aléatoires devant utiliser des graines, excluant donc la fonction *Math.random()*, et dans un environnement contraint comme pour les mobiles (même si les progrès dans ce domaine sont impressionnants), le Central Randomizer de Hoole peut être un compromis intéressant et facile à mettre en œuvre.

3.5 La bibliothèque C standard

On en arrive enfin à un LCG valable, celui décrit par Kernighan et Ritchie dans l'ouvrage de référence sur le C (1978), et qui continue à être présent dans plusieurs implémentations des compilateurs C/C++.

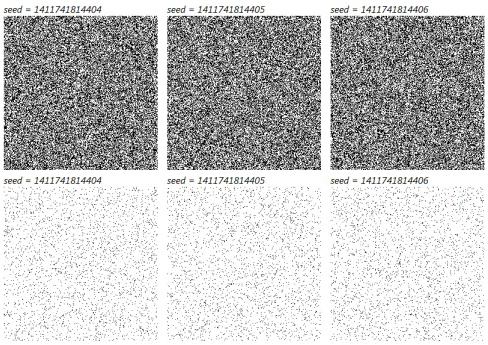
Ce LCG est caractérisé par le triplet $<1103515245, 12345, 2^{16}>$ et par la suite :

— $X_{n+1} = (1103515245 * X_n + 12345)$ **mod** 2^{16}

```
1  var randomSeed = Date.now();
2
3  function randCLib(): number {
4    randomSeed = randomSeed * 1103515245 + 12345;
5    randomSeed = (randomSeed / 65536) % 32768; // extract bits 30..16
6
7    return randomSeed / 32767.0;
8  } // randCLib
```

À noter qu'à l'heure actuelle, des variantes plus sophistiquées de cet algorithme sont implémentées dans les compilateurs C/C++.

On notera que pour la grande majorité des graines, les résultats visuels sont satisfaisants.



Représentations pile ou face et nuages de points du générateur de C

3.6 Bilan des LCG

Le LCG du langage C est-il donc la solution à notre besoin ? Disons que cela dépend. Si on fait abstraction de la cryptographie qui nécessite des générateurs autrement plus complexes, la simulation numérique et à fortiori les jeux vidéo, notamment ceux utilisant intensivement la génération procédurale et les processus stochastiques (e.g. phénomènes physiques, IA), la quantité requise de nombres aléatoires fait que fatalement ce LCG, comme n'importe quel autre, biaisera d'une façon ou d'une autre les nombres aléatoires générés, et ceci en vertu d'un théorème affirmant de façon schématique que les nombres générés par les LCG n'occupent pas tout

l'espace des possibles, mais uniquement une fraction. Le LCG de Windows en est un exemple flagrant, mais le LCG de la bibliothèque C comporte également ce défaut, même si c'est dans une bien moindre mesure.

Par conséquent, pour avoir des nombres aléatoires occupant de façon homogène l'ensemble des possibles il faudra faire appel à d'autres générateurs un peu plus élaborés que les LCG.

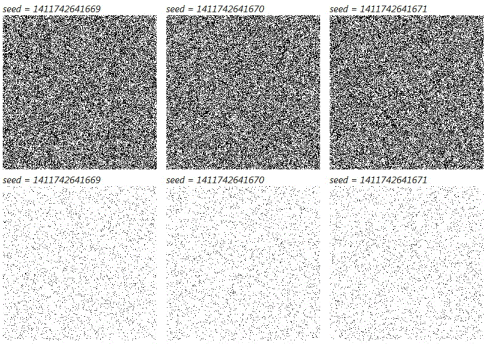
4 Xorshift

Le Xorshift (lien 141) est un générateur aléatoire mis au point en 2003 par George Marsaglia, qui utilise l'opérateur ou exclusif (*xor*) et le décalage de bits (*shift*). Comme les LCG, il a l'avantage d'être simple et rapide, mais n'a pas tous les défauts d'un LCG, même s'il n'est pas parfait.

L'implémentation de cet algorithme pourrait donner ceci en TypeScript :

```
1  var randomSeed = Date.now();
2
3  // Xorshift initialization
4  var x = 123456789;
5  var y = 362436069;
6  var z = 521288629;
7
8  function randXorshift(): number {
9    var t = (x ^ (x << 11)) & 0x7fffffff;
10   x = y;
11   y = z;
12   z = randomSeed;
13   randomSeed = (randomSeed ^ (randomSeed
         >> 19) ^ (t ^ (t >> 8)));
14   return randomSeed / 2147483648.0;
15 } // randXorshift
```

En termes de résultats visuels, rien ne permet de distinguer une série générée par Xorshift de celle générée par le véritable hasard.



Représentations pile ou face et nuages de points du Xorshift

Des études scientifiques ont démontré que cet algorithme est supérieur aux LCG. Il représente donc un meilleur choix en cas d'utilisation intensive. Je soupçonne d'ailleurs Firefox d'avoir implémenté le Xorshift pour la fonction JavaScript standard *Math.random()*. Néanmoins, la version de base du Xorshift présentée ici comporte quelques défauts qu'il serait hors de propos d'aborder ici. Retenons juste que pour pallier ces défauts, il existe une variante du Xorshift, nommée Xorshift* (lien 142), qui est considérée par les spécialistes comme l'un des plus rapides générateurs de nombres aléatoires de haute qualité.

5 Mersenne Twister

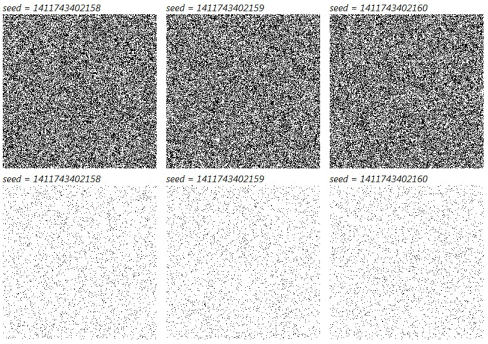
Venons-en à l'un des plus populaires générateurs de haute qualité, le Mersenne Twister (lien 143), mis au point en 1997 par Makoto Matsumoto et Takuji Nishimura. Il doit son nom au fait que cet algorithme se base sur les propriétés des nombres premiers de Mersenne. Pour ce qui est d'une explication de l'algorithme, cela dépasse les compétences de votre humble serviteur, mais il faut savoir que ce générateur est celui utilisé par défaut dans de nombreux langages comme Python, Ruby ou PHP.

Voici une implémentation possible de cet algorithme en TypeScript :

```
1  var randomSeed = Date.now();
2
3  var N = 624;
4  var M = 397;
5  var MATRIX_A = 0x9908b0df; /* constant
       vector a */
6  var UPPER_MASK = 0x80000000; /* most
       significant w-r bits */
7  var LOWER_MASK = 0x7fffffff; /* least
       significant r bits */
8  var mt = new Array(N); /* the array for
       the state vector */
9  var mti = N + 1; /* mti==N+1 means mt[N]
        is not initialized */
10
11 /* initializes mt[N] with a seed */
12 mt[0] = randomSeed >>> 0;
13 for (mti = 1; mti < N; mti++) {
14   var s = mt[mti - 1] ^ (mt[mti - 1] >>>
         30);
15   mt[mti] = (((((s & 0xffff0000) >>> 16)
         * 1812433253) << 16) + (s & 0x000
       0ffff) * 1812433253)
16   + mti;
17   /* See Knuth TAOCP Vol2. 3rd Ed. P.106
         for multiplier. */
18   /* In the previous versions, MSBs of
        the seed affect */
19   /* only MSBs of the array mt[]. */
20   /* 2002/01/09 modified by Makoto
        Matsumoto */
```

```
21    mt[mti] >>>= 0;
22    /* for >32 bit machines */
23  } // for mti
24
25  function randMersenne(): number {
26    var y;
27    var mag01 = new Array(0x0, MATRIX_A);
28    /* mag01[x] = x * MATRIX_A for x=0,1 *
          /
29
30    if (mti >= N) { /* generate N words at
           one time */
31      var kk;
32
33      for (kk = 0; kk < N - M; kk++) {
34        y = (mt[kk] & UPPER_MASK) | (mt[kk
              + 1] & LOWER_MASK);
35        mt[kk] = mt[kk + M] ^ (y >>> 1) ^
              mag01[y & 0x1];
36      }
37      for (; kk < N - 1; kk++) {
38        y = (mt[kk] & UPPER_MASK) | (mt[kk
              + 1] & LOWER_MASK);
39        mt[kk] = mt[kk + (M - N)] ^ (y >>>
              1) ^ mag01[y & 0x1];
40      }
41      y = (mt[N - 1] & UPPER_MASK) | (mt[0
            ] & LOWER_MASK);
42      mt[N - 1] = mt[M - 1] ^ (y >>> 1) ^
            mag01[y & 0x1];
43
44      mti = 0;
45    }
46
47    y = mt[mti++];
48
49    /* Tempering */
50    y ^= (y >>> 11);
51    y ^= (y << 7) & 0x9d2c5680;
52    y ^= (y << 15) & 0xefc60000;
53    y ^= (y >>> 18);
54
55    return (y >>> 0) * (1.0 / 4294967295.0
          );
56  } // randMersenne
```

Comme on s'en doutait, il n'est pas possible de distinguer visuellement une série générée par le Mersenne Twister d'une série réellement aléatoire :



Représentations pile ou face et nuages de points du Mersenne Twister

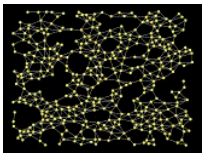
Malgré la haute qualité des nombres aléatoires générés, le Mersenne Twister a un défaut non négligeable qui est une relative lenteur. Pour cette raison, il n'est pas forcément pertinent d'utiliser aveuglément cet algorithme dans des situations où les performances sont cruciales. Dans une telle situation, on lui préférera par exemple le Xorshift ou ses variantes.

6 Utilisation de la graine

Maintenant que nous avons un premier panorama de différents générateurs, on peut s'atteler à leur utilisation dans une application, en se donnant la possibilité de paramétrer la graine si besoin est. Ceci afin de répondre à l'exigence de reproductibilité que nous nous étions fixée en introduction.

En effet, à partir d'une même valeur de graine, la séquence du générateur sera toujours identique dans le cadre des algorithmes présentés qui sont tous déterministes. Il suffit donc de mémoriser la graine pour reproduire à l'envi une même séquence.

Si on s'appuie sur l'exemple de générateur aléatoire d'univers présenté dans un autre sujet (lien 144), une même graine peut donc reproduire un même univers. D'une certaine façon, cela peut être considéré comme une forme de compression.



Univers généré à partir de la graine 1411743524410

Une démonstration en ligne du générateur d'univers est disponible ici : lien 145.

Dans certaines circonstances, la graine peut donc être mise à profit dans des jeux pour économiser de l'espace mémoire. Il est évidemment moins coûteux de stocker un seul nombre, plutôt que des milliers, voire des millions, dans le cas des mondes ouverts.

7 Distribution normale

Les générateurs présentés ici produisent des nombres aléatoires qui suivent la distribution uniforme. C'est-à-dire qu'en théorie, chaque valeur possible a la même probabilité d'apparition. Or, il est assez fréquent qu'on ait besoin d'une autre distribution, par exemple lorsqu'il s'agit de modéliser des phénomènes naturels ou s'en approchant.

La distribution normale aussi appelée distribu-

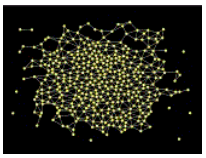
tion de Gauss est la plus utilisée, pour la simple raison que c'est en quelque sorte « la distribution des distributions » en vertu du Théorème de la Limite Centrale (lien 146) sur lequel le lecteur est invité à se documenter par lui-même.

Pour produire des nombres aléatoires suivant la distribution normale, il existe tout un tas de méthodes algorithmiques dont la plus répandue est celle s'appuyant sur la méthode de Box-Muller (lien 147) en coordonnées cartésiennes dont voici une implémentation en TypeScript :

```
function randNorm() {
  var y1, y2, rad;

  do {
    y1 = 2 * rand() - 1;
    y2 = 2 * rand() - 1;
    rad = y1 * y1 + y2 * y2;
  } while (rad >= 1 || rad == 0);

  var c = Math.sqrt(-2 * Math.log(rad) /
      rad);

  return y1 * c;
} // randNorm
```

La fonction *rand()* qui apparaît deux fois, doit être une fonction renvoyant un nombre aléatoire flottant compris entre 0 et 1.

Si on applique une distribution normale à notre d'univers de graine 1411743524410, cela donne ceci :



Univers généré à partir de la graine 1411743524410 en distribution normale

8 Conclusion

De par l'importance des nombres aléatoires dans le domaine de la simulation numérique et des jeux vidéo, il n'est pas inutile de s'intéresser un minimum aux générateurs de nombres aléatoires.

Nous avons vu différents types de générateurs de nombres aléatoires. Mais il en existe bien d'autres.

Les LCG sont une bonne première approche, à condition de bien choisir ses coefficients, mais ils ont des limitations intrinsèques que seules d'autres approches peuvent surmonter comme l'algorithme du Xorshift ou le Mersenne Twister. Finalement, le choix d'un générateur est principalement un compromis entre la qualité des nombres aléatoires produits et la rapidité d'exécution de l'algorithme.

Il convient de garder à l'esprit que tous les générateurs présentés dans cet article ne sont pas exploitables dans le domaine de la cryptographie. Il faut pour cela utiliser des routines véritablement spécialisées et validées par des spécialistes. Pour un complément sur le sujet on peut se référer utilement à cette Introduction à la cryptographie : lien 148.

Enfin, si l'exigence de reproductibilité des nombres aléatoires n'est pas requise, il est possible d'obtenir des nombres réellement aléatoires, issus de processus physiques, via des dispositifs matériels comme des puces, des cartes ou des clés USB, ou encore via divers services Web comme le QRNG Service de l'Université Humbolt de Berlin (lien 149), ou celui de la société random.org (lien 150).

Le code source des différents algorithmes présentés est disponible sur mon compte Github : lien 151.

Retrouvez l'article de ***yahiko*** *en ligne : lien 152*

OpenOffice-LibreOffice

Les derniers tutoriels et articles

Base : comment créer des tables et exécuter les requêtes

Pour commencer une base de données, cela débute par des données qui sont stockées dans des tables, et les requêtes sont là pour les exploiter.
Je vais avec ce tutoriel, vous montrer comment créer des tables, des vues et des requêtes.

1 Définition

Base est l'application de gestion des bases de données des suites bureautiques LibreOffice et OpenOffice.

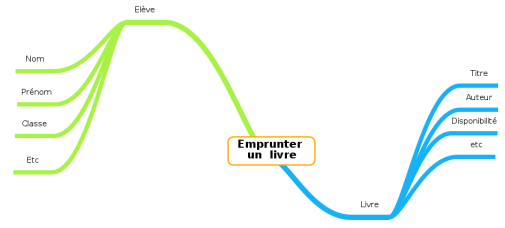
Nous les appelons bases de données relationnelles. Elles sont constituées d'un ensemble de données qui se présente sous la forme d'un tableau, chaque colonne correspond à des informations (des numéros, des dates, des textes plus ou moins longs, etc.), les lignes étant appelées « enregistrements ». Ces tableaux sont appelés des tables, celles-ci sont structurées et doivent avoir le moins de répétitions possible.

Les tables permettent de regrouper toutes les données en un seul endroit, ces dernières pouvant par la suite être utilisées par d'autres programmes.

2 Prérequis

Avant de commencer à concevoir une base de données relationnelle, il est important de bien définir la structure des données, cela afin de créer toutes les tables nécessaires et d'indiquer toutes les relations (notions abordées dans le paragraphe 5) entre elles.

Nous pouvons distinguer deux types de tables : celles qui représentent des entités (informations sur l'emprunteur, les caractéristiques d'un livre, etc.) et celles qui associent une liaison entre deux tables.



Il est important de bien nommer les tables pour permettre de facilement retrouver les informations et aussi d'effectuer une meilleure maintenance de la base de données. Par exemple : pour une table contenant des informations sur des élèves, vous pouvez l'appeler t_info_eleve ou t_eleve, le tout étant de ne pas créer d'ambiguïté sur les informations contenues.

3 Exemple

Nous allons prendre le cas d'une gestion des emprunts des livres dans une école.

La meilleure méthode est d'utiliser un *brainstorming*, la schématisation de notre cas peut être ainsi :

4 Les tables

4.1 Prérequis

Chaque colonne d'une table doit contenir des données de même type et tout changement de structure en cours entraîne des pertes de données.

Le nommage doit lui aussi être significatif, nous permettant par la suite une meilleure maintenance.

Il est aussi recommandé d'utiliser des index (Un index est une donnée qui permet un accès plus rapide à une ligne spécifique d'une table) :

— soit cette valeur est connue, car il existe une donnée unique ;

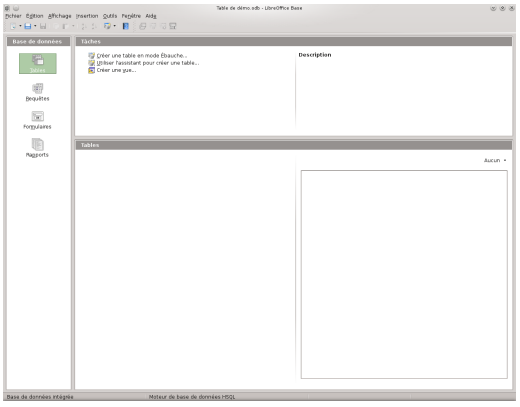
— soit en créant un index automatique.

Quelques recommandations sur les tables et les champs : il est conseillé de ne pas utiliser des mots-clés, d'éliminer les caractères spéciaux, d'écrire tout en minuscule, de remplacer les espaces par un *underscore* « _ ».

Pour le nom des champs, il est important de toujours donner le même nom à un champ, cela n'en sera que plus pratique pour faire la liaison entre les tables.

Il est aussi important de conserver une uniformité dans le nommage des champs, par exemple tous les champs « date » peuvent commencer par « dt ».

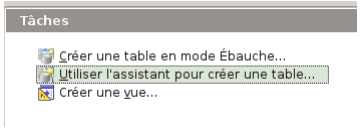
Pour tous les exemples qui suivent, je pars du principe que nous créons une nouvelle base et que nous obtenons donc la fenêtre suivante :



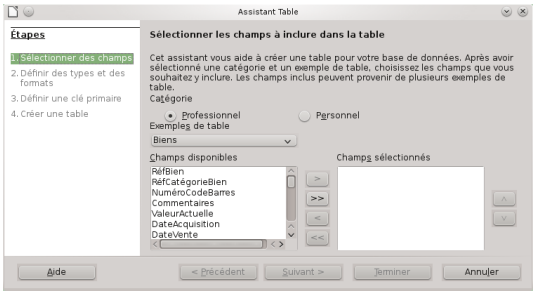
Commençons à créer les tables !

4.2 Avec l'assistant

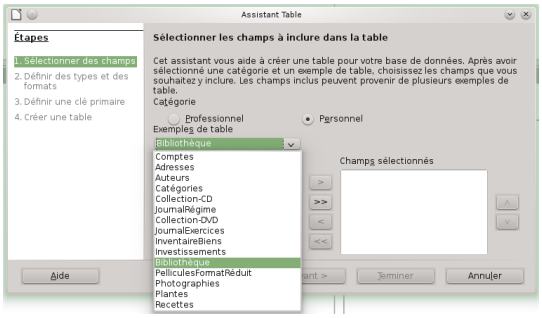
Pour lancer l'assistant, il suffit de cliquer sur :



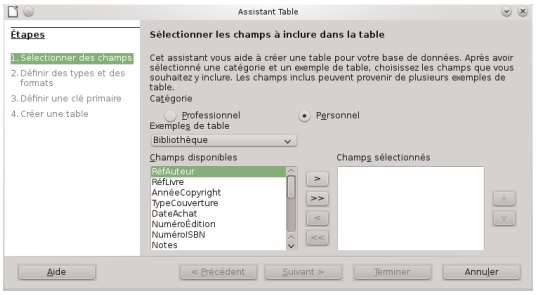
La fenêtre suivante va s'ouvrir :



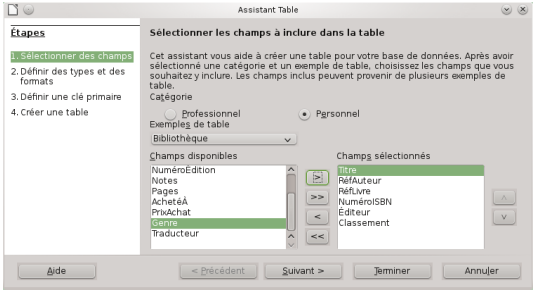
Il nous suffit d'abord de choisir les exemples de tables qui dépendent de la catégorie. Restons sur notre exemple de bibliothèque et sélectionnons « Personnel » et « Bibliothèque » :



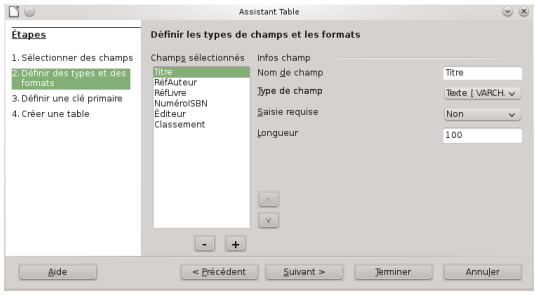
Ensuite, sélectionnons les champs voulus et leur position dans la table avec l'aide des flèches. Nous obtenons en sélectionnant quelques-unes des colonnes :



En cliquant sur « Suivant », la fenêtre ci-après s'affiche :



Il est alors possible, sur cette fenêtre, de modifier les informations d'un champ, qu'il s'agisse du nom ou du type de données.

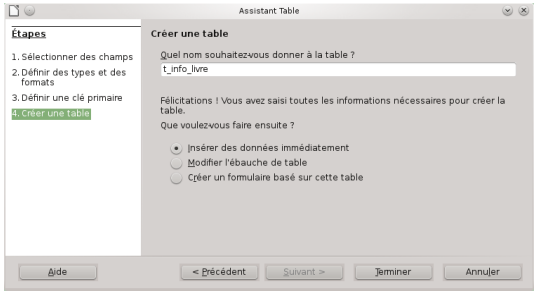


Avec le « + », nous pouvons rajouter des champs. Cliquons sur « Suivant » :

Nous arrivons sur le paramétrage de la clé primaire. Cette dernière est obligatoire pour pouvoir insérer des données. S'il n'y pas de champ unique, nous créons une clé primaire automatique.

Une clé primaire numérique est mieux qu'une clé primaire type texte.

Dans notre cas, nous allons choisir une clé primaire automatique, cliquons sur « Suivant » :



Nous arrivons sur la dernière étape de la création de notre table, n'oubliez pas les prérequis donnés dans le paragraphe précédent.

Nous avons aussi la possibilité de modifier les informations ou de créer un formulaire de saisie.

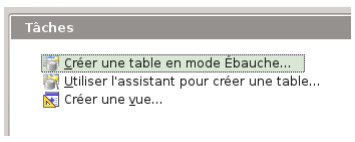
Cliquons sur « Terminer », notre table apparaît :

Il suffit de l'ouvrir pour insérer des données.

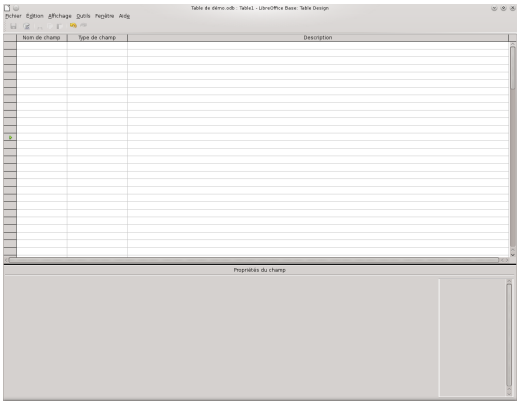
Pour créer toutes vos tables, il vous suffit de procéder ainsi.

4.3 En mode ébauche

Pour cela, il suffit de cliquer sur :



La fenêtre suivante va s'ouvrir :



Nous nous positionnons sur la première ligne, et dans la colonne « Nom de champ », il faut saisir le nom que nous souhaitons lui donner.

Ensuite, dans la colonne « Type de champ », nous allons, à l'aide de la liste déroulante, sélectionner un type de données, la valeur par défaut est un champ texte de dix caractères.

Suivant le choix fait, cela aura pour effet d'actualiser la partie « Propriétés du champ ».

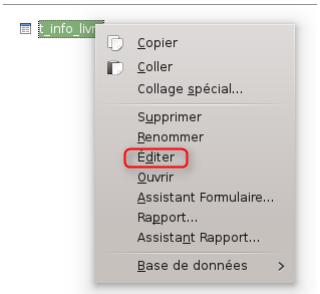
Par exemple, j'ai choisi de créer le champ id_eleve et de le mettre en « integer » autoincrément.

Il vaut mieux commencer par les champs les plus importants comme l'id et les champs que vous utiliserez souvent. *Dans notre exemple, le nom est plus important que la date de naissance.*

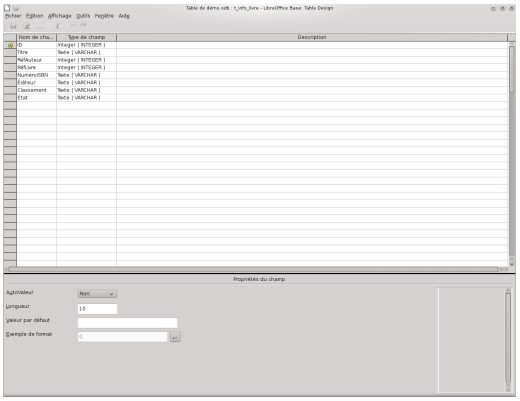
Vous devez procéder ainsi pour tous les champs qui constitueront votre table, mais vous aurez toujours la possibilité de revenir dessus pour modifier les éléments. Il faut savoir que la modification d'un type de données peut entraîner la perte de certaines informations.

Modification d'une table

Pour modifier une table, nous devons la sélectionner et faire un clic droit. Nous obtenons :



Et en cliquant sur « Éditer », nous affichons la fenêtre suivante :



Nous arrivons sur le même masque que dans la création en mode ébauche. Il nous suffit ensuite de faire les modifications voulues.

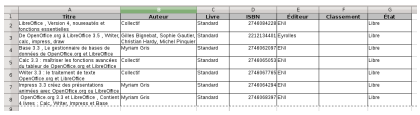
Si vous changez le type de données d'une colonne et que celle-ci contient des données, vous risquez de les perdre.

Les champs que vous ajouterez à la table seront automatiquement ajoutés à la fin.

4.4 Import

Pour cet exemple, nous prendrons le cas d'un fichier Tableur (Calc), mais cela fonctionne de la même façon pour tout fichier *csv*, *xls*, *xlsx*, etc.

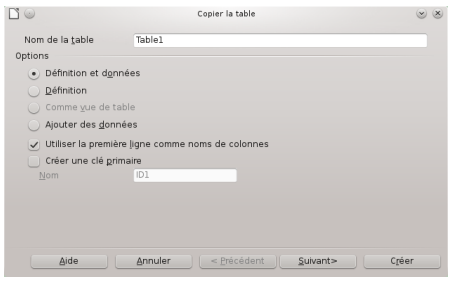
Le fichier a la forme suivante :



L'insertion des données se fait à l'aide du copier/-coller. Il nous faut donc sélectionner les cellules de A1 à G8 et les copier (Ctrl+C ou dans le menu « Édition/Copier »).

4.4.a Table et données

Ensuite, il nous suffit d'aller dans le module des tables et de faire le collage (Ctrl+V ou dans le menu « Édition/Coller »), la fenêtre suivante apparaît :



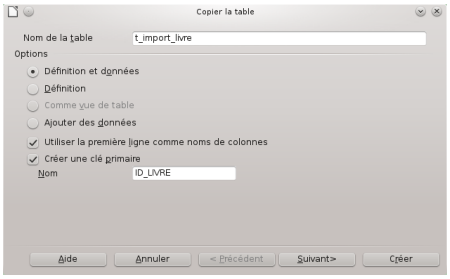
Sur ce module, il nous faut :

— nommer la table ;

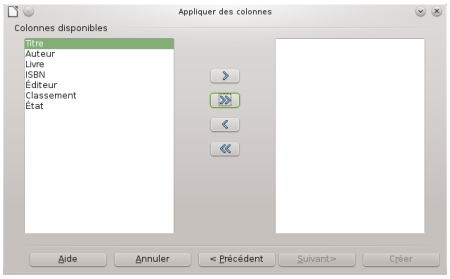
— vérifier que la commande « Utiliser la première ligne comme noms de colonne » soit bien activée ;

— vérifier que la commande « Créer une clé primaire » soit bien activée et la nommer.

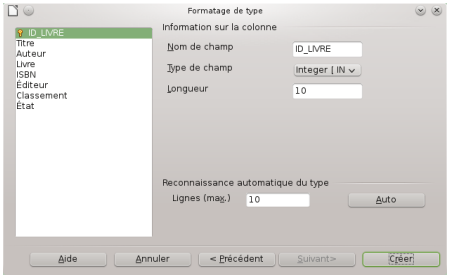
Ce qui nous donne finalement :



Cliquons sur « Suivant », la fenêtre devient :

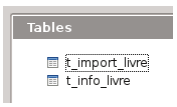


Il nous suffit de sélectionner les champs que nous souhaitons avoir. Après cela, le bouton « Suivant » apparaît, cliquez dessus, la fenêtre suivante s'affiche :



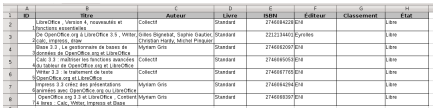
Il nous faut contrôler si les types de champs correspondent à ce que nous souhaitons mettre dans la colonne. Par défaut, il est tenu compte des données pour définir le champ.

Pour finir la création, il faut cliquer sur « Créer », et la table se crée :

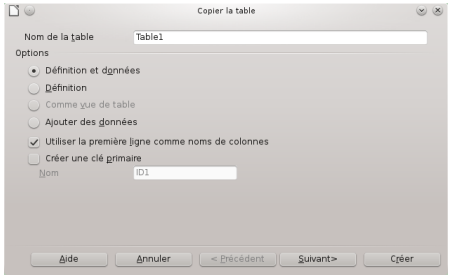


4.4.b Données seulement

Pour insérer les données dans une table, celles-ci doivent contenir autant de champs ou colonnes. Dans notre cas, il nous manque le champ ID, il nous faut le rajouter :



Ensuite, nous allons procéder comme précédemment. Sélectionnons les cellules A1 à G8, faisons une copie (Ctrl+C ou dans le menu « Édition/Copier »), et faisons un collage (Ctrl+V ou dans le menu « Édition/Coller ») dans la base de données, la fenêtre suivante apparaît :

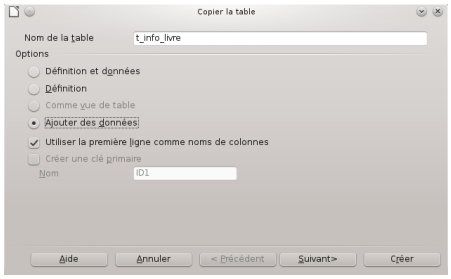


Sur ce module, il nous faut d'abord :

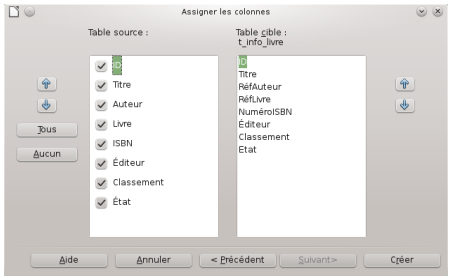
— mettre dans le champ « Nom de la table » le nom de la table où nous souhaitons inclure les données ;

— vérifier que la commande « Ajouter les données » soit bien activée.

Ce qui nous donne :



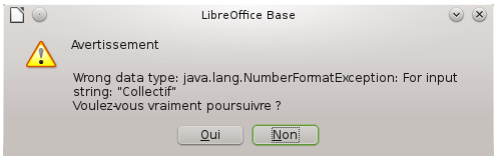
Ensuite, cliquons sur « Suivant », la fenêtre suivante apparaît :



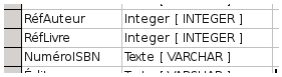
Maintenant, il faut contrôler que les champs de la table « source » correspondent bien à ceux de la table cible. Il suffit de sélectionner un des champs pour voir à qui il est associé.

Une fois la vérification faite, cliquons sur « Créer ».

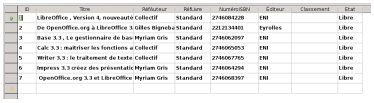
Si un des champs contient une donnée ne pouvant pas être incluse dans la table, vous verrez apparaître le message suivant :



Dans notre cas, certains champs n'ont pas le bon type de données :

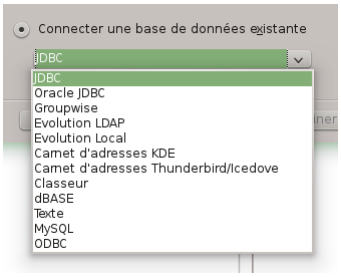


Une fois les types de données en cohérence avec les données à insérer, nous obtenons dans la table :

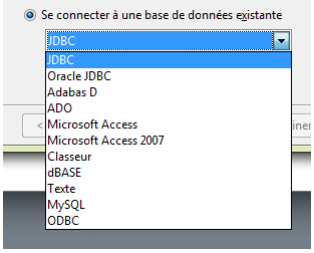


4.5 Se connecter avec des données externes

La connexion avec des données externes dépend de l'environnement et de l'application utilisée, comme vous pouvez le voir sur ces exemples :



LibreOffice sous OpenSuse



OpenOffice sous Windows 7

Il existe un tutoriel pour vous connecter à une base MySQL : « LibreOffice Calc requêter une base MySQL » (lien 153).

5 Les relations

5.1 Définition

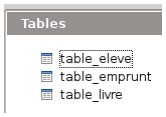
Une relation est le lien qui existe entre deux tables, associant des données de la première avec celles de la seconde.

Nous rencontrerons trois situations, les relations pourront être :

— d'un enregistrement à un seul autre : 1 à 1
— d'un enregistrement à plusieurs autres : 1 à ∞ ou 1 à n
— de plusieurs enregistrements à plusieurs autres : ∞ à ∞ ou n à n

5.2 Créer des liens entre les tables

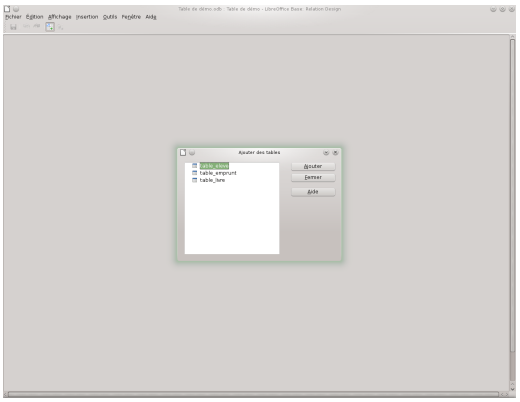
Prenons comme exemple celui-ci :



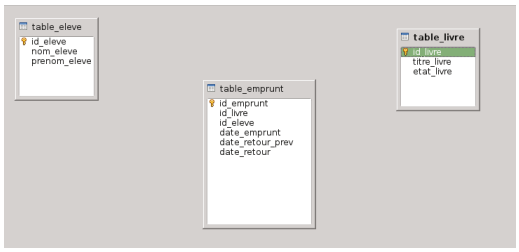
Ces tables ont des relations entre elles.

La commande se trouve dans le menu « Outils » et « Relations... » :

La fenêtre suivante s'ouvre :

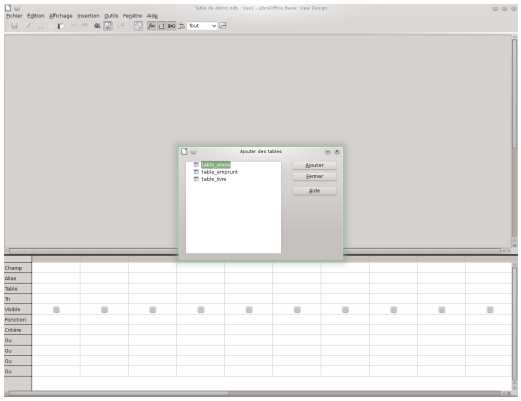


Il suffit d'ajouter les tables pour lesquelles nous souhaitons définir des relations, ce qui nous donne :



Un bon agencement des tables permet une meilleure lisibilité des données par la suite.

Pour définir les relations, il suffit de sélectionner une des données et de la faire glisser sur la donnée correspondante sur une autre table. Dans notre cas, nous obtenons :



Il ne faudra pas oublier d'enregistrer avant de sortir de cette fenêtre.

Maintenant, lorsque vous mettrez deux tables qui ont une relation entre elles dans une requête, la liaison se fera automatiquement.

6 Les requêtes

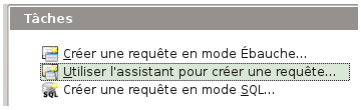
Les requêtes permettent d'interroger, d'ajouter ou de modifier les données de la base. Elles peuvent être triées et filtrées.

Commençons par créer des requêtes !

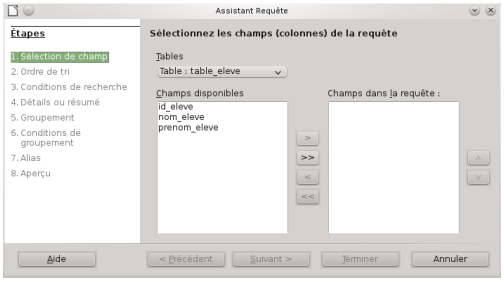
Ce mode de création de requête n'est pas adapté si vous souhaitez utiliser plusieurs tables.

6.1 Avec l'assistant

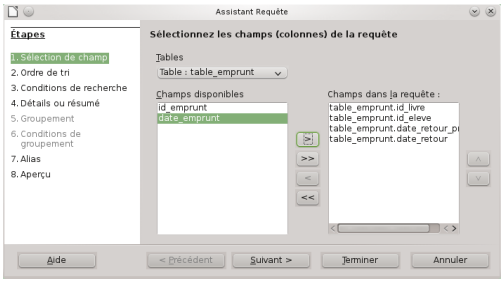
Pour lancer l'assistant, il suffit de cliquer sur :



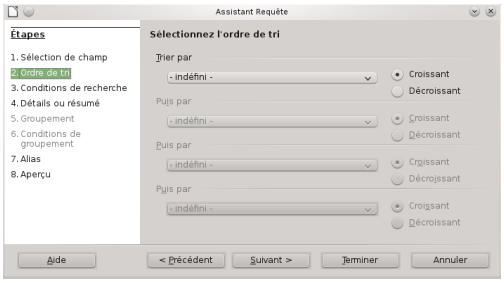
La fenêtre suivante va s'ouvrir :



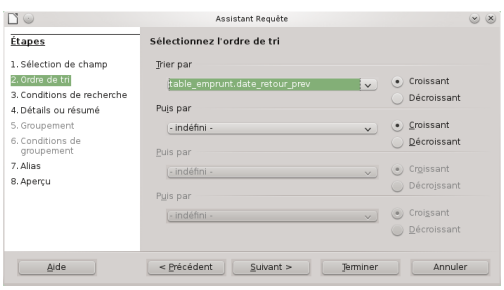
Il faut sélectionner la table que nous souhaitons interroger et les champs dont nous avons besoin, ce qui nous donne par exemple :



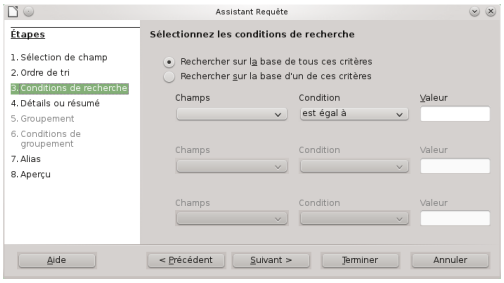
En cliquant sur « Suivant », la fenêtre suivante apparaît :



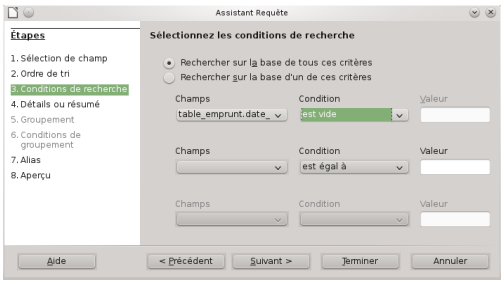
Il nous suffit de sélectionner les tris que nous souhaitons faire et le sens (croissant ou décroissant), par exemple en triant les dates de retour prévues :



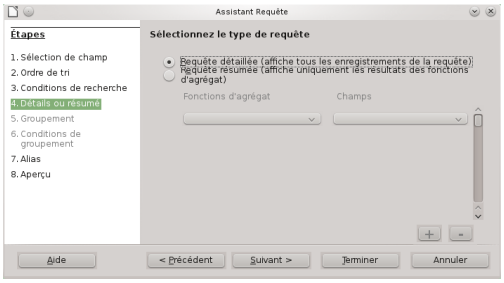
En cliquant sur « Suivant », la fenêtre suivante apparaît :



Il nous suffit de mettre les conditions que nous souhaitons avoir, par exemple en ne sélectionnant que les emprunts qui n'ont pas de date de retour :

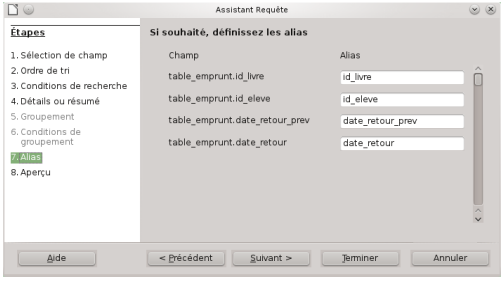


En cliquant sur « Suivant », la fenêtre suivante apparaît :



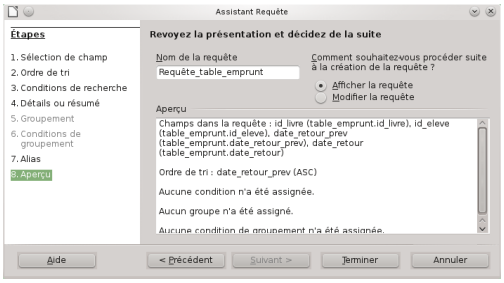
Cette partie permet de faire des regroupements, cela est utile si vous souhaitez faire des sommes, des moyennes...

En cliquant sur « Suivant », la fenêtre suivante apparaît :

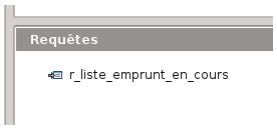


Il nous est possible de changer les noms des champs, cela peut être utile dans le cas où ces derniers ne sont pas explicites. Par exemple : date_retour_prev pourrait être date de retour prévu du livre.

En cliquant sur « Suivant », la fenêtre suivante apparaît :



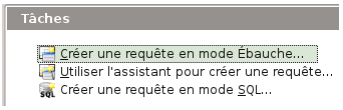
Sur ce module, vous devez nommer votre requête, ensuite vous avez un résumé de toutes les actions que vous voulez faire sur cette requête. En cliquant sur « Terminer», la requête est créée :



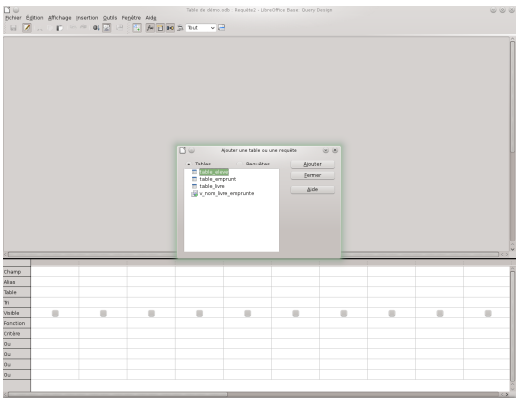
Pour la suite, la requête pourra être modifiée.

6.2 En mode ébauche

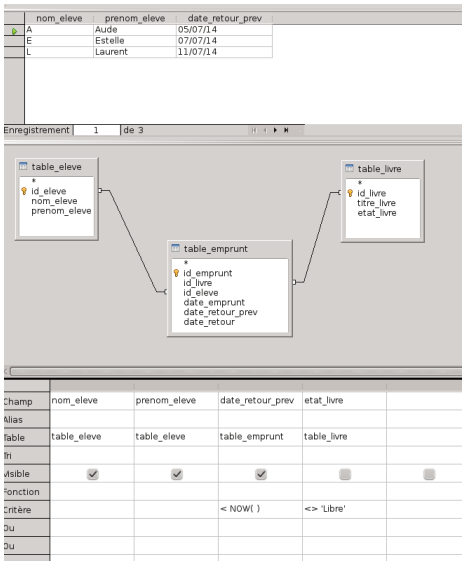
Pour cela, il suffit de cliquer sur :



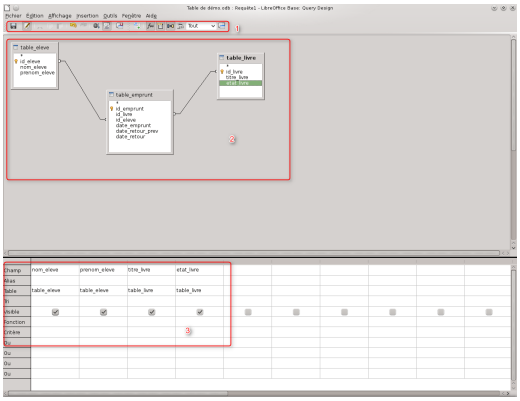
La fenêtre suivante va s'ouvrir :



La fenêtre suivante permet d'ajouter les tables ou les requêtes dont nous aurons besoin. Il nous suffit de les sélectionner, de cliquer sur « Ajouter » puis sur « Fermer » une fois tous les éléments sélectionnés :



Nous arrivons ensuite sur une fenêtre qui ressemble à cela :



— C'est la barre d'outils qui contient tous les éléments :

a : contient les éléments d'enregistrement et d'édition ;

b : contient le copier / coller, etc.

c : contient l'affichage des données, le changement de mode d'affichage, etc.

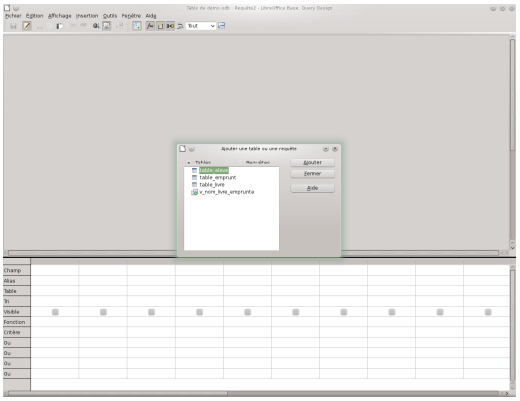
d : permet d'ajouter des tables ou des requêtes ;

e : contient des éléments permettant de modifier la partie qui se trouve au bas de l'écran.

— C'est le bloc qui contient les tables et les requêtes, et surtout les liens qui connectent les éléments entre eux.

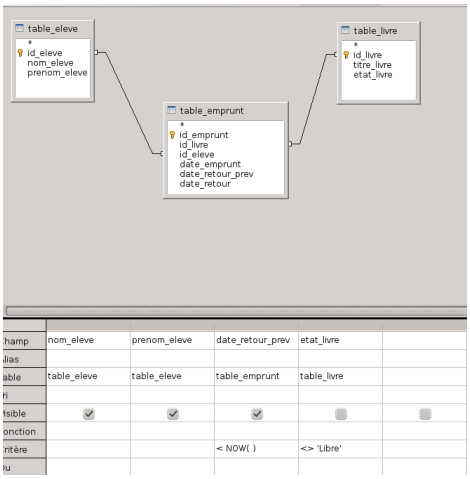
— C'est la partie qui liste les éléments que nous voulons voir apparaître. C'est aussi là que nous pouvons mettre des filtres, des tris et faire des calculs.

Dans notre cas, nous obtenons ceci, une fois les tables ajoutées, les liens ajoutés ou modifiés et les champs ajoutés ainsi que les tris et les filtres souhaités :

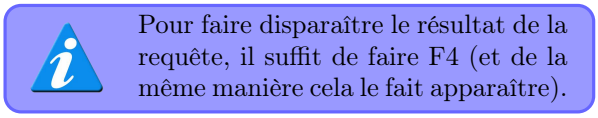


Dans l'exemple ci-dessus, nous souhaitons récupérer les noms et les livres qui sont empruntés, mais dont la date de retour est dépassée.

En cliquant sur :

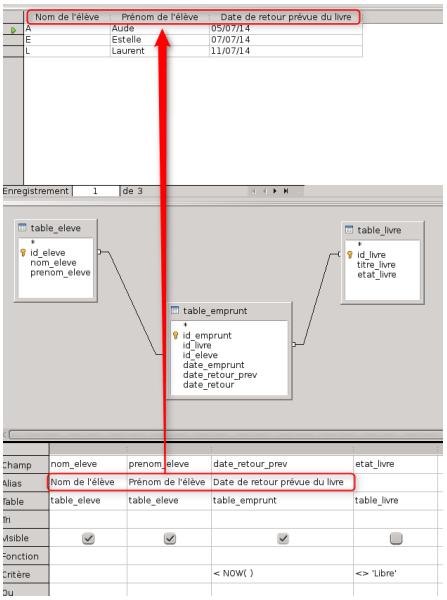


vous avez un aperçu du résultat :

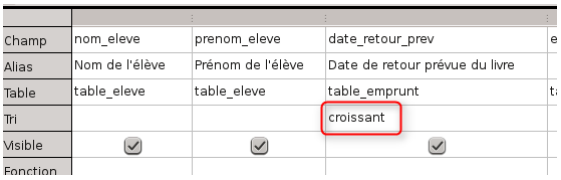


Pour faire disparaître le résultat de la requête, il suffit de faire F4 (et de la même manière cela le fait apparaître).

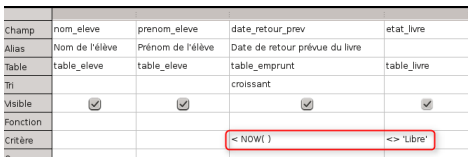
Il est possible de renommer les titres des colonnes et de voir le résultat, il suffit de faire F5 pour actualiser :



Nous voulons maintenant que les dates de retour prévues soient classées par ordre croissant. À l'aide de la liste déroulante, nous sélectionnons l'ordre voulu, ce qui nous donne :

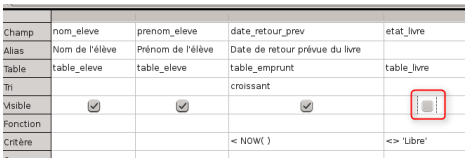


Nous pouvons aussi rajouter des filtres sur certaines colonnes, en saisissant le filtre choisi :



S'il n'y a pas de signe avant le filtre, cela signifiera « égal ».

Maintenant, nous ne souhaitons voir que les trois premières colonnes, il nous faut donc décocher la visibilité, ce qui nous donne :

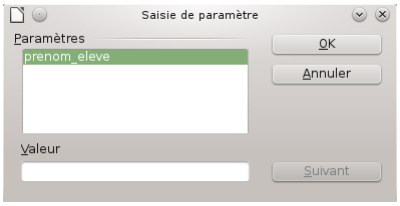


Ce qui nous manque dans notre vue, c'est par exemple, le nombre de jours de retard. Nous allons donc rajouter un champ calculé, il nous faut nous positionner dans le champ et saisir la formule correspondante. Voici l'aperçu :

La commande SQL correspondante est la suivante : DATEDIFF('dd', "table_emprunt"."date_retour_prev", NOW( ))

Pour la suite, nous avons aussi la possibilité de mettre un filtre paramétré dans le critère, il suffit alors de mettre un « ? » dans le champ :

Lors de l'exécution de la requête, le masque suivant apparaîtra :

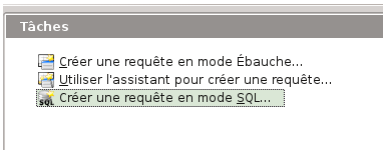


Il ne nous reste plus qu'à saisir la valeur. S'il y avait plusieurs paramètres, ils apparaîtraient les uns au-dessous des autres.

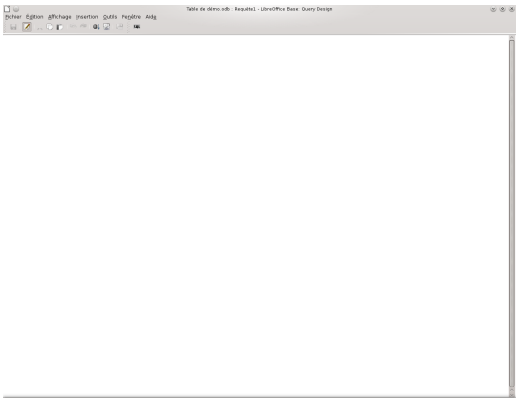
La valeur saisie doit être exactement la même. Si vous cherchez « Laurent » et que vous avez saisi « laurent », cela ne ramènera rien.

6.3 En SQL

Pour cela, il suffit de cliquer sur :

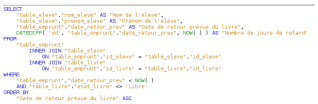


La fenêtre suivante va s'ouvrir :



Il ne nous reste plus qu'à saisir le code SQL sur la page blanche. Vous avez ici tous les éléments pour connaître les commandes SQL : Les meilleurs tutoriels SQL (lien 154)

Si nous reprenons notre cas précédent, l'écriture SQL donne (j'ai pris soin de bien détailler le code) :

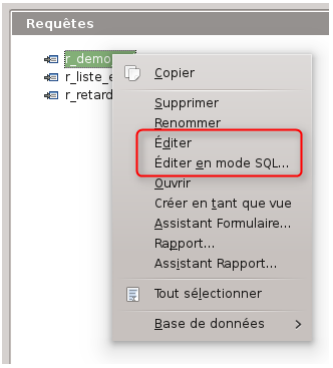


Petite particularité : les noms des tables et des champs doivent être entre guillemets.

Ne passez pas trop de temps sur la mise en forme de la requête. En effet, celle-ci est modifiée par la suite, voici le code qui est créé :

6.4 Modifier une requête

Pour cela, il suffit de faire un clic droit sur la requête :



Suivant votre choix, vous pourrez ouvrir la requête en mode graphique ou SQL.

Si vous vous êtes trompé, vous pourrez toujours, à l'aide du bouton de la barre d'outils, passer de l'un à l'autre :

7 Les vues

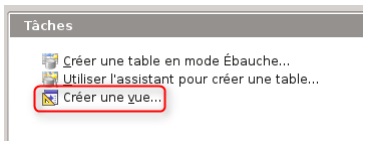
7.1 Définition

Une vue est similaire à une requête, elle peut être aussi appelée table virtuelle.

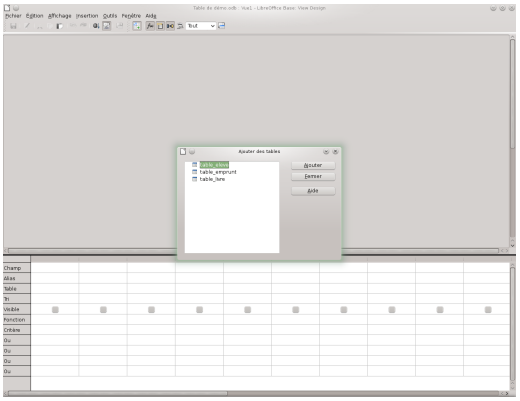
Les cas d'emploi d'une vue sont pour des requêtes souvent utilisées ou pour sécuriser des informations SQL aux utilisateurs. En effet, les protections d'une vue ne sont pas les mêmes que celles d'une table.

7.2 Création

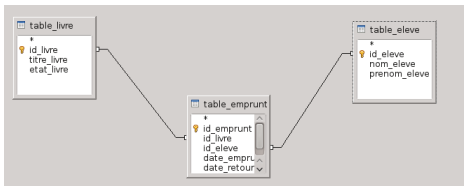
Pour créer une vue, nous cliquons sur la commande suivante :



Ce qui nous ouvre cette fenêtre :



Nous sélectionnons la ou les tables nécessaires en les ajoutant :

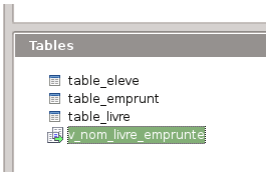


Ensuite, il ne reste plus qu'à sélectionner les champs nécessaires et mettre les filtres souhaités :

Champ	nom_eleve	prenom_eleve	titre_livre	date_retour_prev	etat_livre
Alias					
Table	table_eleve	table_eleve	table_livre	table_emprunt	table_livre
Tri					
Visible	☑	☑	☑	☑	☐
Fonction					
Critère					<> 'Libre'

Dans notre exemple, nous ne souhaitons récupérer que les livres indisponibles, en affichant qui les a empruntés et quand est prévu leur retour.

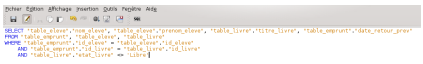
Nous devons l'enregistrer, ce qui nous donne finalement :



Les vues ne fonctionnent que pour des sélections.

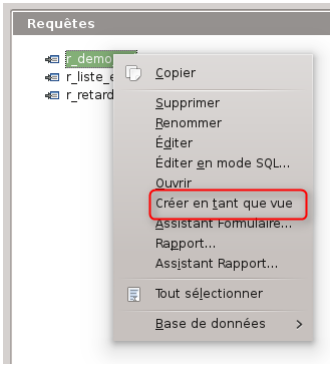
Vous pouvez passer en mode SQL en cliquant sur le bouton suivant :

L'affichage deviendra :



7.3 Complément

Si vous aviez créé une requête, vous pouvez la transformer en vue avec la commande suivante :



Il vous sera alors demandé de la nommer :

Si vous utilisez ce procédé pour créer des vues, il faut savoir que les filtres paramétrés ne fonctionnent pas.

Retrouvez l'article de ***Vincent Viale*** *en ligne : lien 155*

Base : comment créer et modifier des formulaires

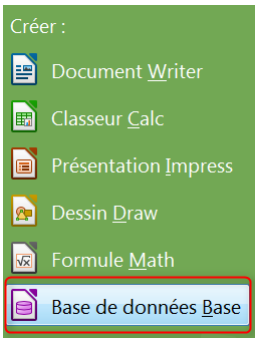
Je vais vous montrer, au cours de ce tutoriel et avec des exemples, comment créer, modifier et personnaliser des formulaires.

1 Introduction

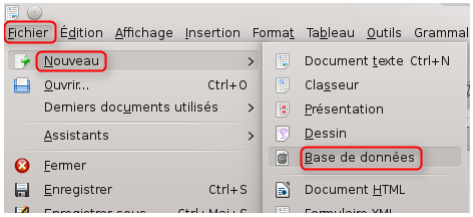
Base est la solution de création et de gestion de bases de données de la suite bureautique LibreOffice/OpenOffice. Afin de consulter, ajouter, modifier ou supprimer les données stockées dans les tables, les formulaires de saisie ou de consultation constituent des interfaces ergonomiques entre l'utilisateur et les tables. Nous allons voir comment créer et personnaliser des formulaires au travers d'exemples simples et en privilégiant l'aide des assistants Formulaire.

2 Créer une base de données

Une fois que LibreOffice (ou OpenOffice) est ouvert, nous avons deux possibilités pour créer une nouvelle base :

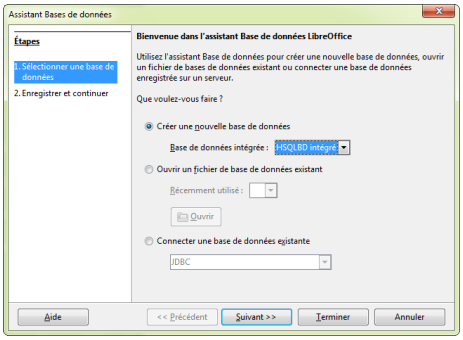


Avec le lanceur

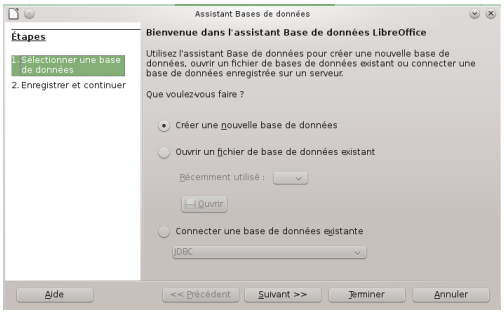


Avec le menu

Les deux affichent la même fenêtre, mais celle-ci peut contenir des éléments différents :

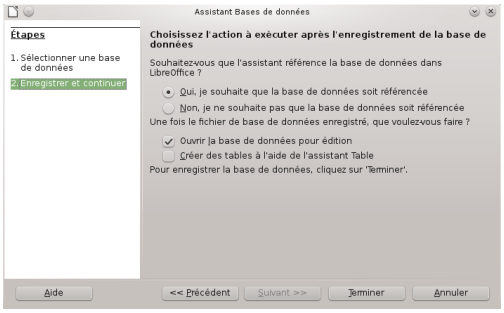


Exemple sous Windows

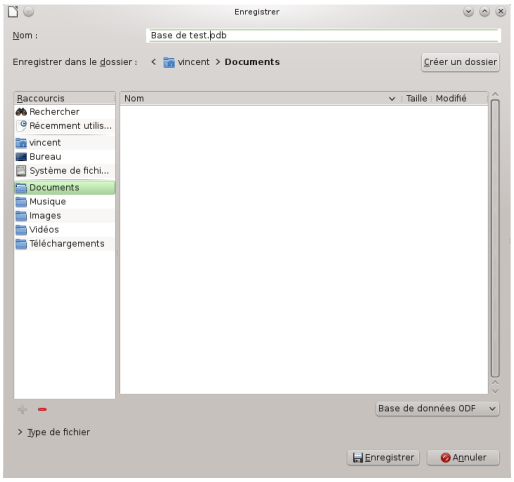


Exemple sous OpenSuse

Ensuite nous cliquons sur « Suivant », cette fenêtre s'ouvre :

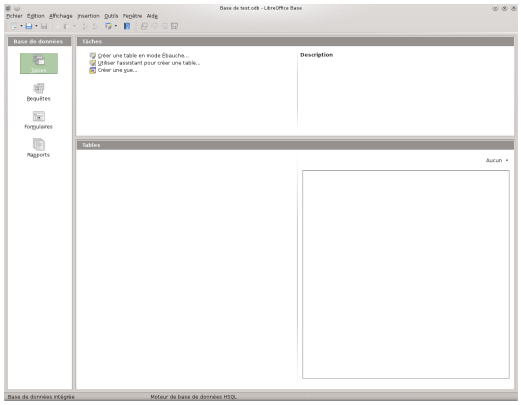


Et nous finissons par « Terminer », ce qui affiche cette fenêtre :



Une base de données doit toujours être enregistrée avant de commencer à y travailler. *Ce qui n'est pas le cas pour les autres programmes de la suite bureautique.*

Une fois que nous avons donné un nom à notre base, l'enregistrement est fait, nous arrivons sur cette fenêtre :

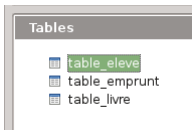


Nous y retrouvons tous les éléments qui composent une base de données : les tables, les requêtes, les formulaires et les rapports.

Pour l'exemple, je vais partir sur le cas d'emprunt de livres, et je vais créer trois tables :

- table_livre : qui contiendra toutes les infos sur les livres ;
- table_eleve : qui contiendra toutes les infos sur les élèves ;
- table_emprunt : qui contiendra tous les éléments sur les emprunts.

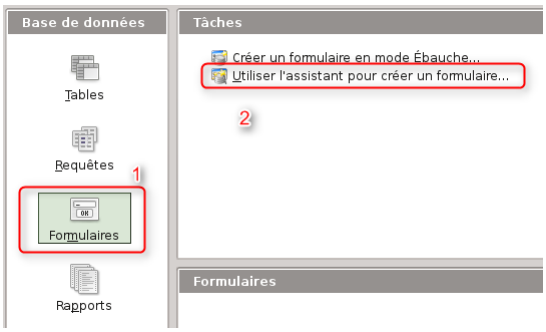
Ce qui nous donne :



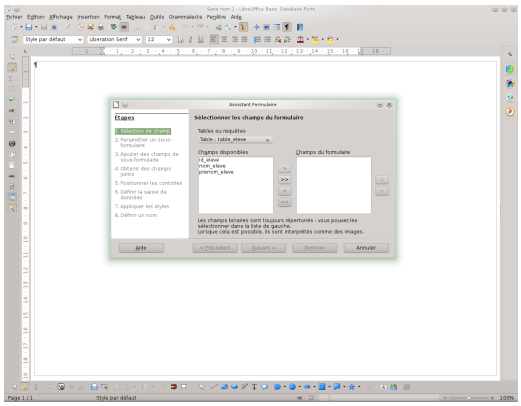
3 Créer un formulaire simple

Nous allons prendre l'exemple d'un formulaire qui nous permettra de voir la liste et l'état des livres d'une bibliothèque.

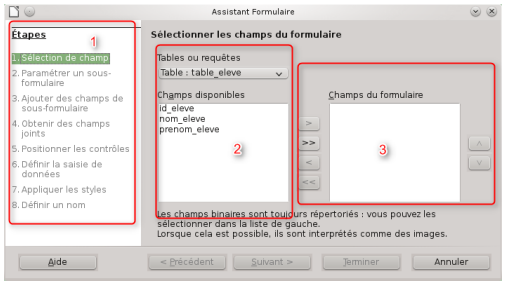
Nous allons utiliser l'assistant pour cela :



Ce qui va ouvrir la fenêtre suivante :

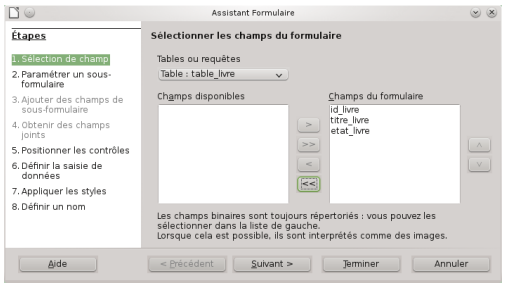


Sur la fenêtre nous pouvons distinguer trois parties importantes :

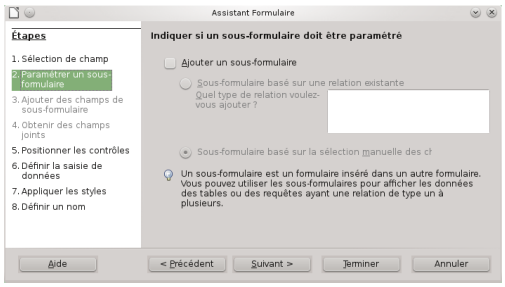


1. La liste des étapes qui vont nous permettre de créer un formulaire ;
2. La liste des données disponibles, s'il y avait des vues et des requêtes, elles seraient aussi visibles ;
3. Cette partie contiendra tous les éléments que nous voulons voir dans le formulaire en fonction de la table ou requête sélectionnée.

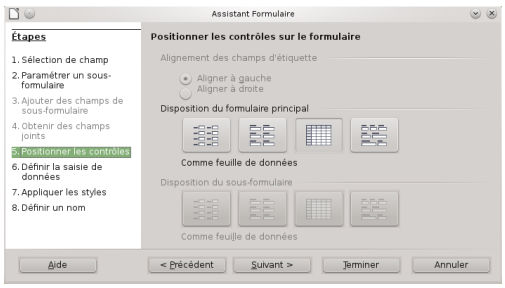
Dans notre cas prenons la table livre :



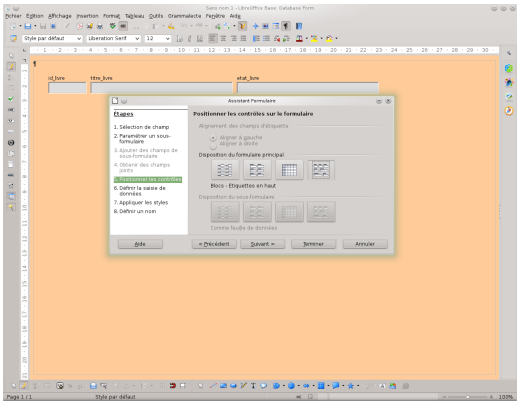
Ensuite, il nous suffit de cliquer sur « Suivant » :



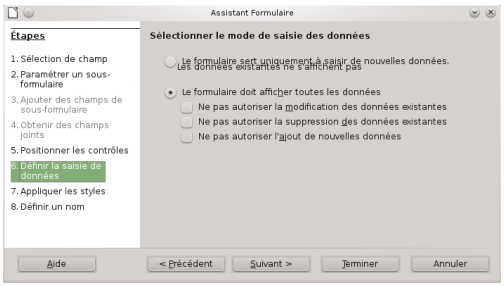
Comme nous sommes dans la construction d'un formulaire simple, cliquons sur « Suivant », ce qui nous donne :



Si nous sélectionnons un modèle, cela l'applique :



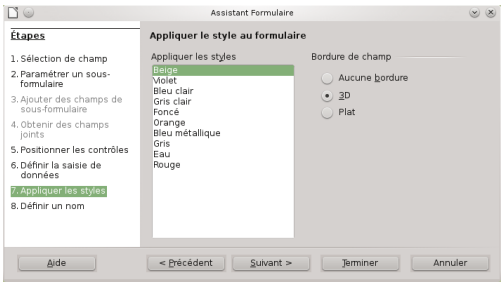
Une fois que nous avons choisi notre modèle, cliquons sur « Suivant » :



Sur cette fenêtre, nous pouvons voir qu'un formulaire peut servir à plusieurs choses :

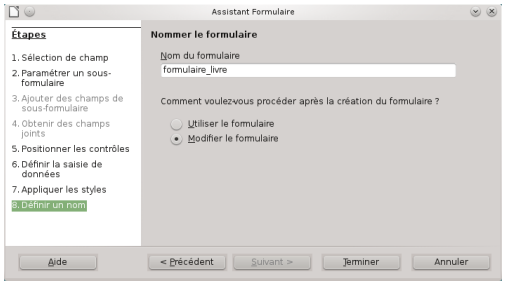
— uniquement à ajouter des données ;
— uniquement à voir les données ;
— à voir les données et à en ajouter.

Une fois votre choix fait, cliquez sur « Suivant », la fenêtre suivante apparaît :

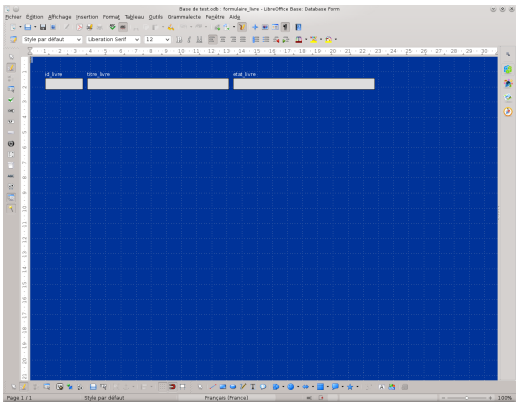


Cette fenêtre permet de modifier l'apparence, il suffit de sélectionner un style et le type de bordure :

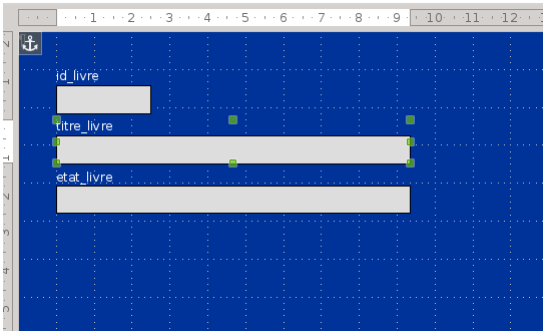
Une fois que vous avez fait votre choix, cliquez sur « Suivant » :



Dans notre cas, prenons l'option « Modifier le formulaire » et cliquons sur « Terminer », la fenêtre suivante s'affiche :



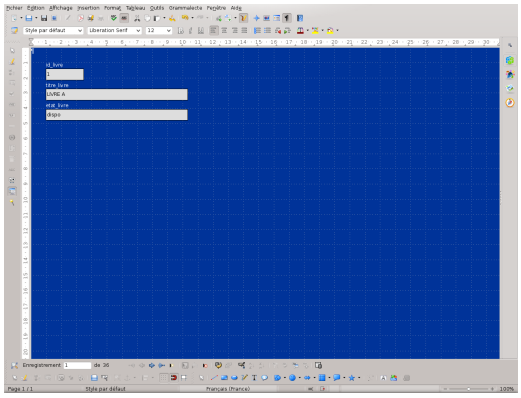
Nous pouvons déplacer les éléments, il suffit de les sélectionner et de les faire glisser :



Nous avons la barre d'outils qui va nous permettre de faire différentes modifications :

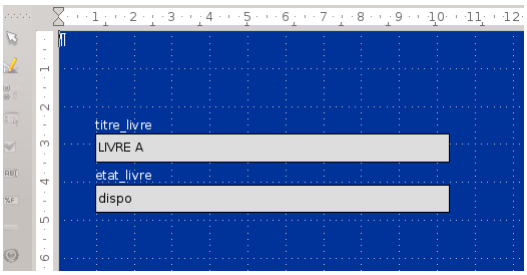
1. Permet de passer en mode ébauche ou pas ;
2. Permet de modifier les propriétés d'un champ ;
3. Permet de modifier les propriétés du formulaire ;
4. Permet de rajouter des champs dans le formulaire.

Une fois que le « mode ébauche » n'est plus actif, nous obtenons :



Maintenant, nous allons masquer le champ id_livre qui ne nous est pas utile. Pour cela, il faut le sélectionner et cliquer sur  , ce qui va ouvrir la fenêtre suivante :

Il ne nous reste plus qu'à mettre le champ « Visible » sur « Non », et une fois le « mode ébauche » désactivé, nous obtenons cette fenêtre :



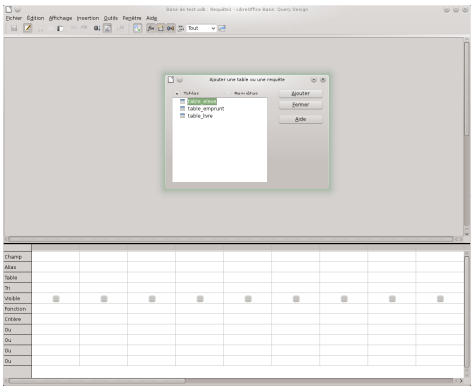
4 Formulaire avec sous-formulaire

Nous allons toujours rester dans la gestion documentaire, mais cette fois nous allons rajouter les informations sur l'emprunt quand il y en a.

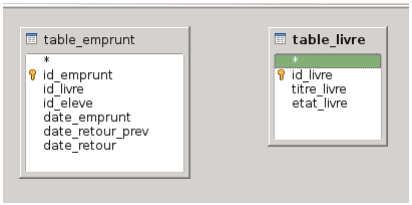
4.1 Création d'une requête

Pour ce cas « *plus complexe* », nous allons créer une requête en mode ébauche, à partir de la fenêtre

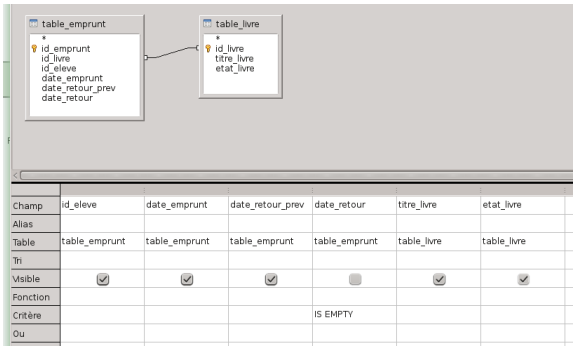
suivante :



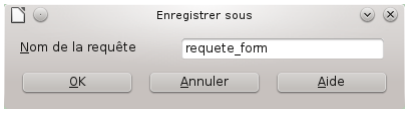
Nous n'avons plus qu'à sélectionner les tables table_emprunt et table_livre, ce qui nous donne :



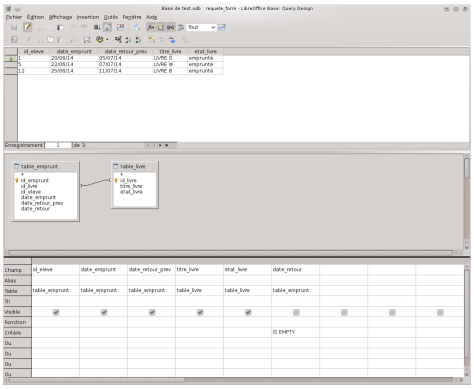
Et maintenant, nous allons créer les relations entre les tables et faire apparaître les éléments souhaités :



Dans notre cas, la relation est l'id_livre, et nous ne voulons voir que les livres qui sont empruntés, c'est pour cela que nous avons mis « is empty » dans le critère du champ date_retour. Ensuite, enregistrons la requête :

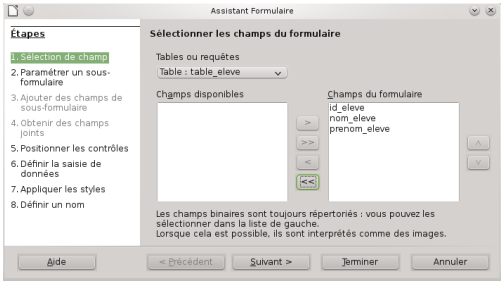


Et voici le résultat de la requête :

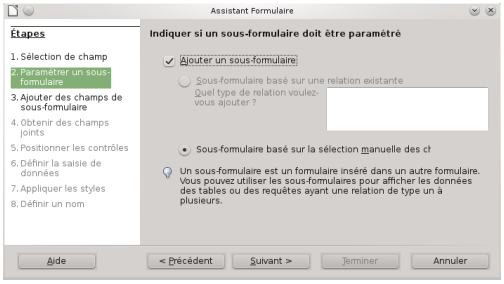


4.2 Création du formulaire avec sous-formulaire

Nous procéderons comme pour le formulaire précédent en utilisant l'assistant, et en sélectionnant la table table_eleve :

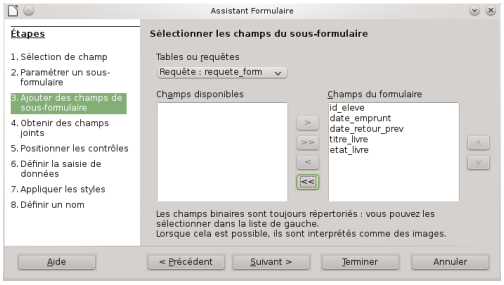


Cliquons sur « Suivant » et nous obtenons la fenêtre suivante :

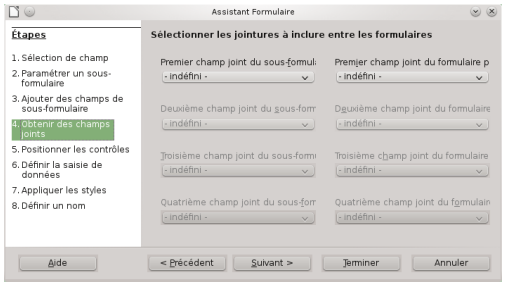


Si nous avions défini des relations dans notre base, celles-ci seraient visibles dans le champ « Sous-formulaire basé sur une relation existante ».

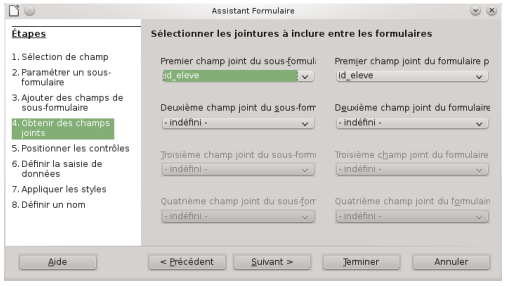
Il nous suffit de cocher « Ajouter un sous-formulaire » et de cliquer sur « Suivant » :



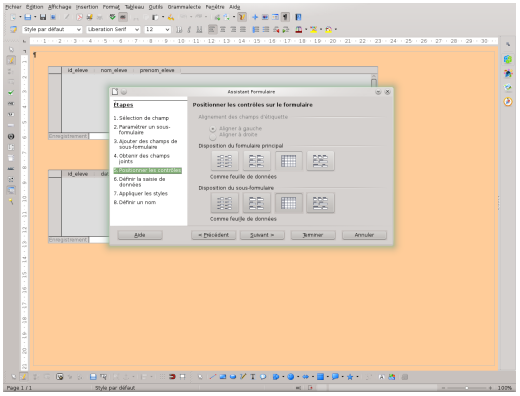
Dans la fenêtre qui apparaît, il suffit de sélectionner la requête que nous avons créée précédemment, ainsi que tous les champs, ensuite cliquons sur « Suivant », ce qui nous donne :



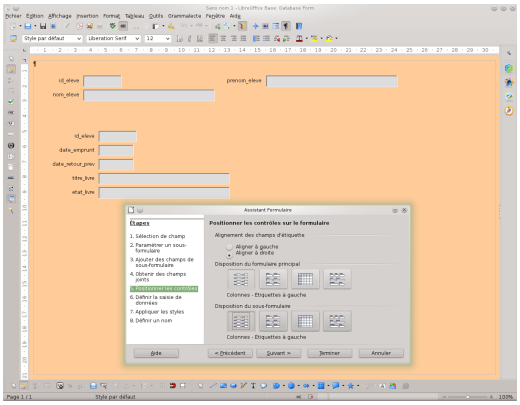
Il va falloir établir au moins une jointure entre le formulaire et le sous-formulaire, dans notre cas, cela sera l'id_eleve :



Cliquons sur « Suivant », la fenêtre suivante apparaît :



Il ne nous reste plus qu'à sélectionner le type d'agencement que nous souhaitons avoir :



Les masques suivants sont identiques à ceux du 2. Nous obtenons finalement :

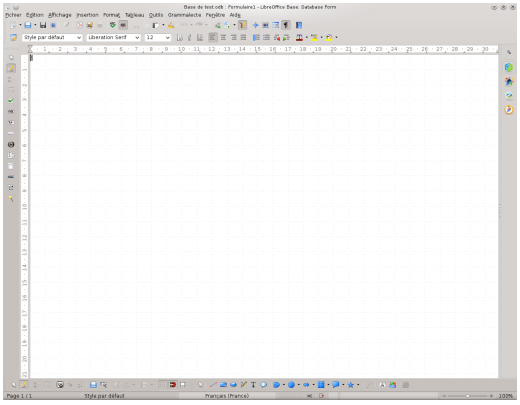
Si l'élève a emprunté un livre

Si l'élève n'a pas emprunté de livre

Pour se déplacer dans les enregistrements, il faut utiliser les commandes :

5 Formulaire en mode ébauche

5.1 Créer un formulaire



Nous reprendrons le même cas que dans le paragraphe 2.

Nous allons cette fois cliquer sur « Créer un formulaire en mode Ébauche... », nous arrivons sur la page suivante :

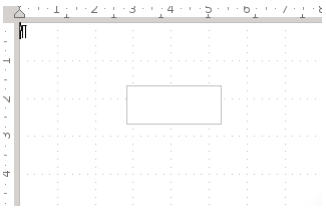
Maintenant comme nous pouvons le voir, nous ne pouvons pas affecter de table (ou requête) à notre formulaire, car le bouton n'est pas actif :

Suivant le thème et l'application les icônes ne sont pas les mêmes. Mais la position sur la barre d'outils reste la même.

Nous allons devoir insérer un champ avant, pour pouvoir définir la table (ou requête), pour cela il suffit de cliquer sur une des commandes possibles pour insérer un champ :

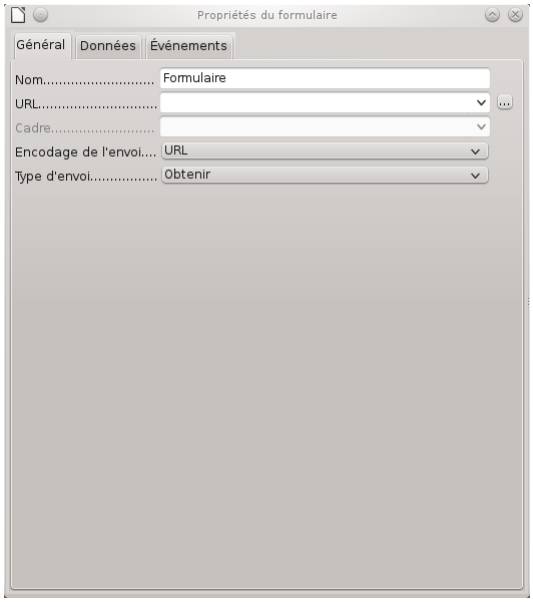
Dans notre cas, nous allons insérer une « Zone de texte » :

Il nous suffit ensuite de créer le champ sur le formulaire en délimitant la zone, nous obtenons alors :

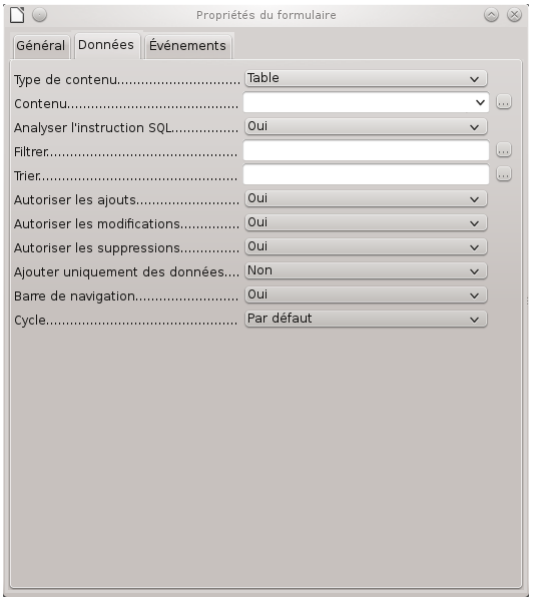


Le bouton devient actif et nous allons pouvoir affecter les données :

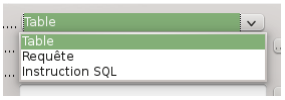
La fenêtre suivante apparaît :



Sélectionnons l'onglet « Données » :

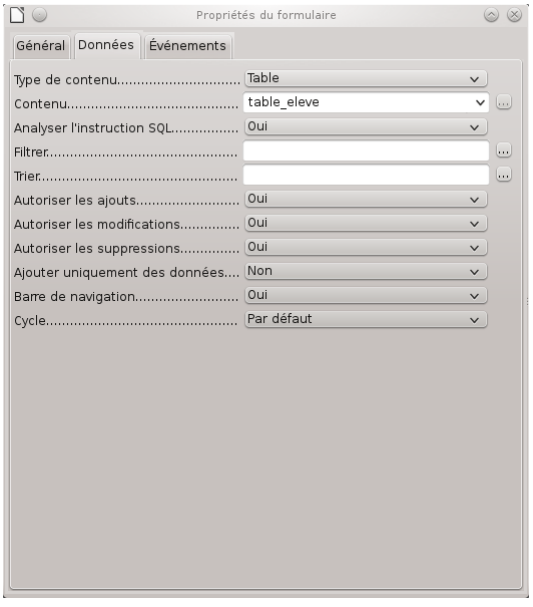


Sur ce masque, nous avons la possibilité de choisir différents types de données :



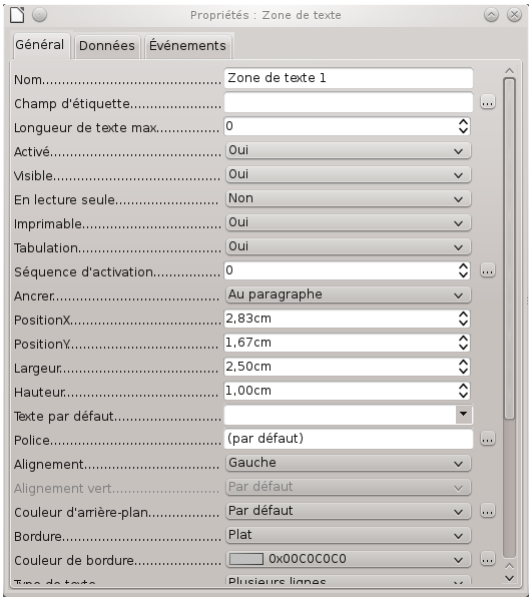
En faisant votre choix, il faut bien faire attention à ce que vous voulez en faire, car si vous sélectionnez Requête ou Instruction SQL, l'ajout et la modification de données seront alors impossibles.

Dans notre cas nous allons choisir « Table » et « table_eleve » :



Ensuite, il ne nous reste plus qu'à sélectionner le champ inséré tout à l'heure et à cliquer sur le bouton qui donne les propriétés du champ :

Ce qui nous ouvre la fenêtre suivante :



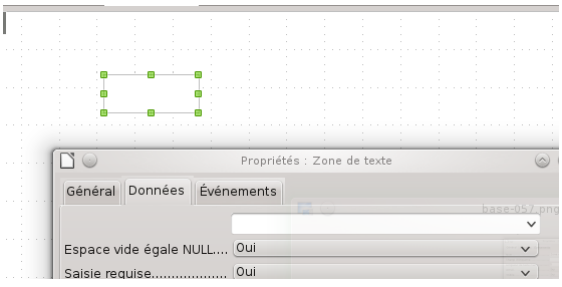
Nous avons trois onglets :

— « Général » concerne les éléments graphiques et le nom du champ, il est important de nommer les champs, cela permet ensuite de les retrouver ;

— « Données » concerne tout ce qui est en rapport avec la valeur qu'aura le champ ;

— « Événements » concerne tout ce qui pourra être ajouté comme action sur le champ, généralement, ce sont des macros qui y sont associées.

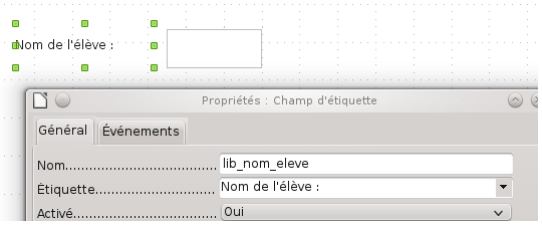
Même si nous y affectons un champ, il nous manque une information :



En effet, nous n'avons qu'un champ vide, nous n'avons aucune indication sur ce à quoi il correspond, c'est pour cela que nous allons maintenant lui affecter une étiquette :

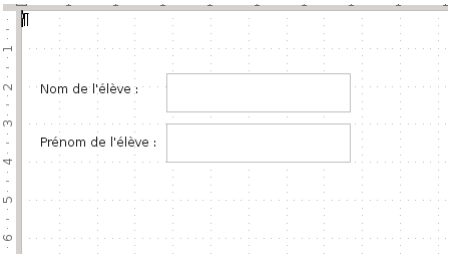
Le procédé reste le même que pour l'insertion de la « Zone de texte », ce qui nous donne :

Il ne nous reste plus qu'à aller dans la propriété de l'étiquette, pour changer le nom de l'étiquette et le texte :



Il est important de bien faire la distinction entre le champ et l'étiquette.

Il faut procéder ainsi pour chaque champ. Nous obtenons le formulaire suivant :



Si vous souhaitez changer la couleur de fond du formulaire, la commande se trouve dans le menu « Format », « Page... » et dans l'onglet « Arrière-plan ».

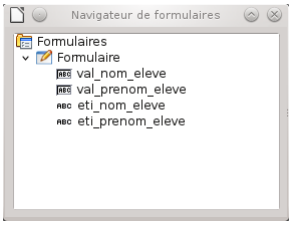
La dernière étape étant l'enregistrement, il suffit de cliquer sur la disquette ou d'aller dans le menu « Fichier ».

5.2 Créer un sous-formulaire

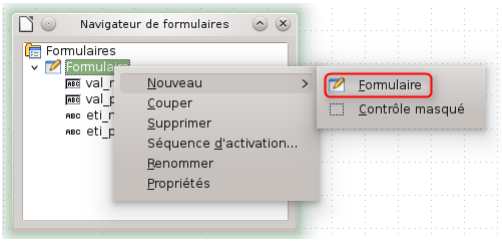
Nous nous trouvons dans le formulaire que nous venons de créer précédemment :

Nous devons ouvrir le « Navigateur de formulaires » :

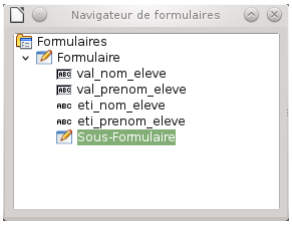
Nous obtenons sur notre page la fenêtre suivante :



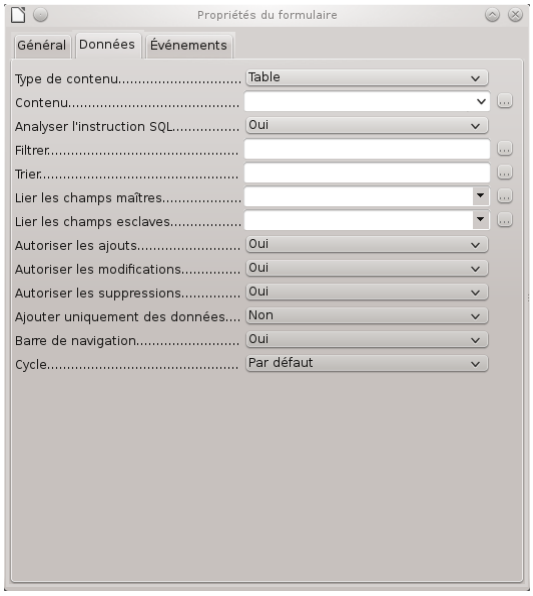
Pour insérer un sous-formulaire, il nous suffit de sélectionner le formulaire actuel dans la fenêtre et de faire un clic droit :



Une fois que nous avons sélectionné l'option, il nous suffit de renommer le sous-formulaire :



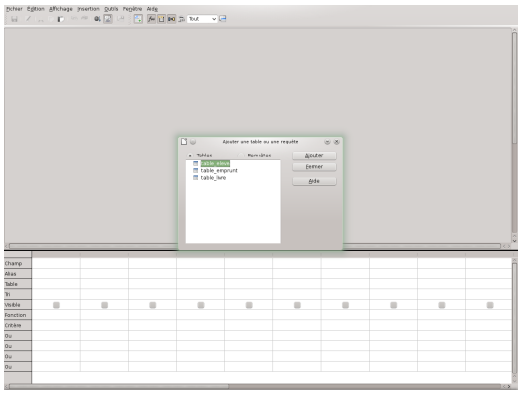
Pour maintenant affecter la table, la requête ou le code SQL au sous-formulaire, il nous suffit de faire un clic droit (« Propriétés ») ou de cliquer sur le bouton correspondant, ce qui nous ouvre la fenêtre suivante :



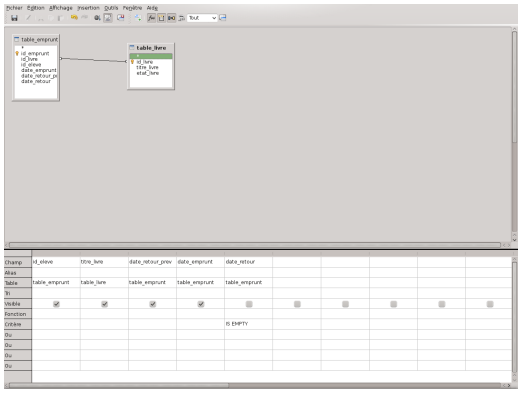
La fenêtre ne contient aucune information, il ne nous reste plus qu'à sélectionner les éléments que nous souhaitons voir apparaître dans ce sous-formulaire.

Dans le « Type de contenu », nous sélectionnons « Instruction SQL » , soit nous connaissons le code SQL correspondant, il n'y a plus qu'à le saisir dans le contenu, soit nous ne le connaissons pas et nous cliquons sur les « ... » :

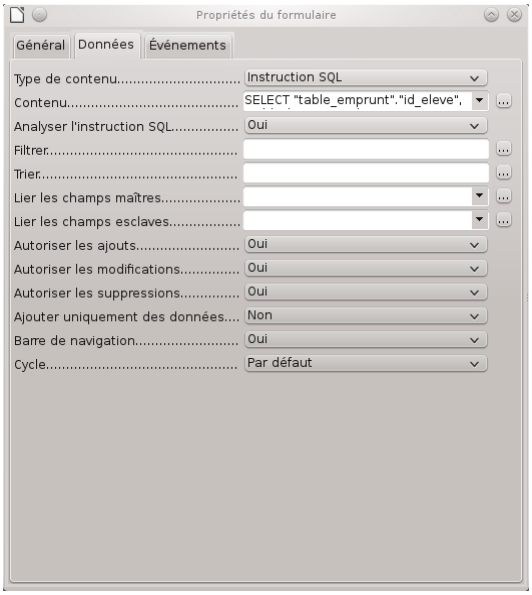
La fenêtre suivante s'ouvre :



Il ne nous reste plus qu'à créer les requêtes voulues :

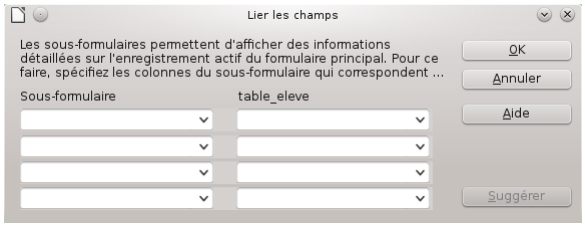


Pour revenir au masque précédent, il suffit de fermer la fenêtre, ce qui nous donne :

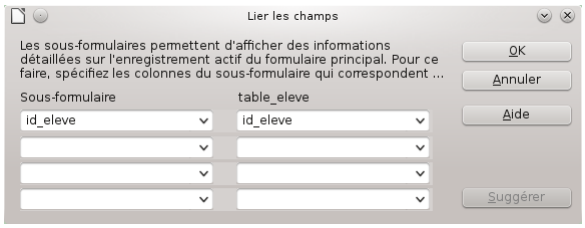


Maintenant, il faut lier le formulaire et le sous-formulaire, les commandes se trouvent ici :

Nous n'avons plus qu'à cliquer sur « ... » pour ouvrir la fenêtre qui nous permettra de créer le (ou les) lien(s) :



À l'aide des listes déroulantes, il nous faut sélectionner les champs en commun entre les deux, ce qui nous donne finalement :

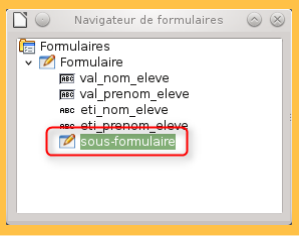


Nous cliquons sur « OK » pour terminer. Nous pouvons dire que la partie récupération des données pour le sous-formulaire est maintenant faite, il ne reste plus qu'à insérer les données sur le formulaire.

Vous pouvez modifier d'autres paramètres sur cette fenêtre, à vous de voir lesquels vous voulez modifier.

Suivant la forme que vous voulez donner au sous-formulaire, le mode opératoire ne sera pas le même :

il faut que la fenêtre « Navigateur du formulaire » soit visible et que le sous-formulaire soit sélectionné :



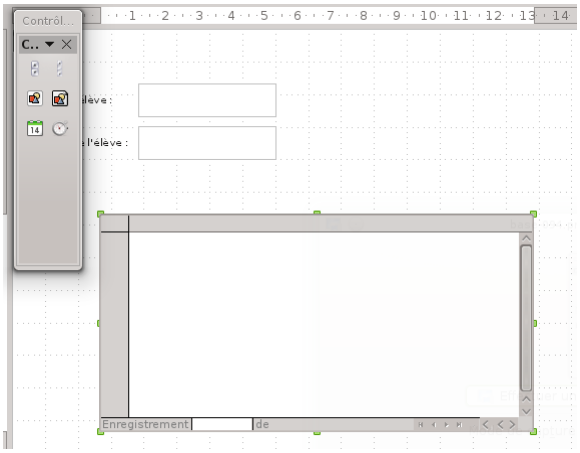
— soit vous voulez ajouter les champs, et la façon de faire reste la même que pour le formulaire ;
— soit vous voulez créer un tableau, il faut pour cela activer les « Contrôles supplémentaires ».

Pour activer les « Contrôles supplémentaires », il faut cliquer sur la commande suivante :

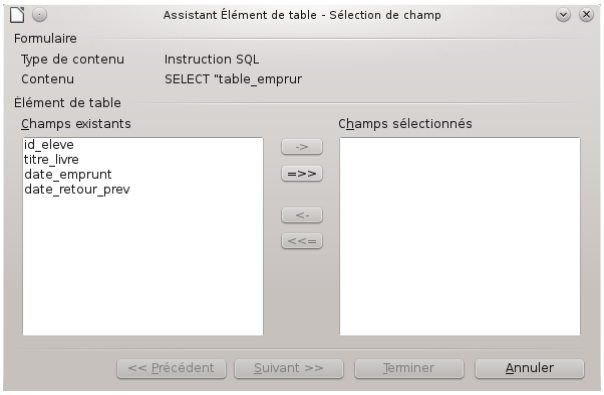
La fenêtre suivante apparaît alors :

Il nous suffit de sélectionner la commande « Contrôle de table » :

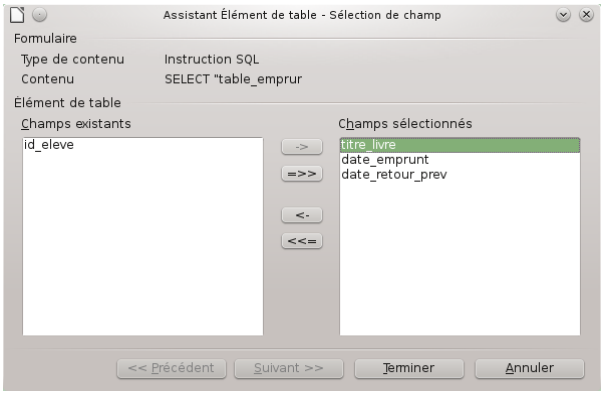
Ensuite sur le formulaire, il reste à délimiter la zone de notre sous-formulaire :



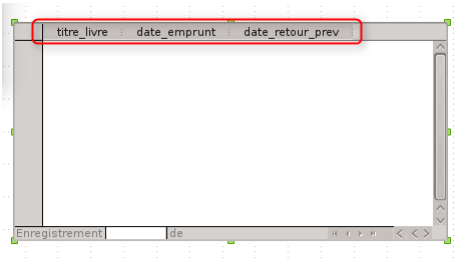
La fenêtre suivante apparaît :



Il ne reste plus qu'à sélectionner les champs que nous souhaitons voir apparaître :



Les champs sélectionnés sont maintenant disponibles dans la fenêtre :



6 Conclusion

Nous venons de voir comment créer des formulaires simples ou avec des sous-formulaires. Maintenant il ne vous reste plus qu'à concevoir vos propres formulaires pour vos applications.

L'utilisation des formulaires permet d'améliorer vos applications en :

- n'affichant que certains éléments ;
- ne mettant en évidence que certains champs avec la modification de l'apparence (couleur, taille, etc.) ;
- aidant à la saisie ;
- la navigation est plus simple que dans des tables ou requêtes ;
- etc.

Retrouvez l'article de ***Vincent Viale*** *en ligne : lien 156*

Qt

Les dernières news Qt

Qt Creator 3.2.0 est disponible

Qt Creator 3.2.0, une nouvelle version mineure, apporte de nombreuses fonctionnalités ainsi que les traditionnels correctifs de bogues. Ci-suit la liste des principales nouveautés :

- dans l'éditeur de texte, vous pouvez maintenant faire de « l'édition en colonne », ce qui consiste à modifier plusieurs lignes différentes en même temps ;
- l'aide contextuelle peut maintenant être configurée pour s'ouvrir dans une autre fenêtre (sans désactiver le mode d'aide) ;
- différents correctifs pour le modèle de code C++ sont disponibles, notamment le support de l'initialisation désignée de C99 (designated initializers), la concaténation de chaînes de caractères, ainsi que des corrections pour les problèmes d'encodage et bien d'autres choses encore ;
- de nouveaux panneaux sont interrogeables, notamment l'arbre de projets ;
- le profileur QML a reçu de nombreux correctifs de stabilité et se trouve être de plus en plus performant.

Vous pouvez aussi jeter un œil au fichier des changements (lien 157), pour une liste plus complète des nouveautés apportées par cette version. Il faut aussi remercier tous les contributeurs, qui furent plus de 50 pour cette version, pour les efforts fournis.

Vous trouverez la version open-source sur la page des téléchargements du projet Qt : lien 158. Merci de faire remonter les problèmes rencontrés sur l'outil de gestion des bogues : lien 159. Vous pouvez aussi trouver les équipes sur IRC, #qt-creator sur irc.freenode.net ainsi que sur la *mailing list* de Qt Creator : lien 160.

Note. Qt Creator 3.2 ne supporte plus OS X 10.6 (Snow Leopard). La raison est qu'Apple ne fournit aucun compilateur compatible avec le standard C++11 pour ce système d'exploitation. Bien sûr, vous pourrez toujours déployer et exécuter vos applications Qt sur cette plate-forme. Simplement, vous ne pourrez ni exécuter Qt Creator 3.2 sur ce système ni le compiler avec la chaîne fournie par Apple : il ne sera plus possible de développer sous cette version, sortie en 2009.

*Commentez la news d'**Arnold Dumas** en ligne : lien 161*

Nouvelle licence pour Qt : LPGL 3

Depuis les débuts de Qt, il y a une vingtaine d'années, l'édition libre existe. Elle fut d'abord limitée aux plateformes UNIX et X11, puis s'est progressivement ouverte, notamment pour Windows et divers systèmes embarqués. De même, la licence a évolué : depuis une licence non standard, de plus en plus libre avec les versions (qui se souvient de la QPL ?), Qt 4 est passé à la GPL 2, ce qui a eu pour effet d'éliminer les conflits juridiques qui empêchaient d'utiliser Qt dans une application GPL, avec une mise à jour vers la GPL 3 dès sa diffusion par la FSF. L'une des principales fonctionnalités de ces licences est d'imposer que les logiciels dérivés (soit tout utilisateur) publient leur code source sous la même licence (« licence virale »). En 2009, Qt 4.5 devient disponible sous la LGPL 2.1 (lien 162) : la licence commerciale n'est plus requise pour un développement fermé, la LGPL n'impose plus la licence sur les produits dérivés.

Cependant, cette version de la LGPL n'est pas parfaite : on peut lui reprocher de n'être pas assez claire au sujet des modifications de la bibliothèque, Qt en l'occurrence. Ainsi, bien qu'en violation totale de l'esprit de la licence, certains constructeurs interdisent l'installation de versions modifiées de la bibliothèque sur leurs appareils. La FSF a corrigé cette faille dans la LGPL 3, qui formalise de manière juridique l'intention première, ce qui protège mieux la liberté des utilisateurs. C'est pour cette raison que Digia proposera Qt 5.4 également sous

licence LGPL 3, en plus de la LGPL 2.1 et de la licence commerciale.

Nouveaux modules

Cependant, certains modules de Qt ne seront disponibles qu'en LGPL 3 et sous une licence commerciale, comme le nouveau Qt WebEngine (pour l'édition libre)(lien 163). D'autres nouveaux modules pourront être libérés (alors que Digia ne les prévoyait que dans l'édition commerciale !), comme Qt Canvas3D (support complet de WebGL à l'intérieur de Qt Quick)(lien 164) ou Qt WebView(intégration assez légère du moteur de rendu Web natif dans Qt, actuellement uniquement pour Android)(lien 165). De même, le style Android pour les Qt Quick Controls sera disponible : il posait problème avec la LGPL 2, puisqu'il utilise des composants sous licence Apache 2.0, une licence incompatible, mais pourra être intégré grâce à la LPGL 3.

Impact sur les utilisateurs de Qt

Tous ceux qui utilisaient Qt sous la licence GPL 3 n'auront aucun souci à se faire, étant donné la compatibilité avec la LGPL 3. Tous les modules de Qt 5.3 seront toujours disponibles sous LGPL 2.1, ce qui ne posera aucun problème. Par contre, si vous mettez à jour vers Qt 5.4 et utilisez l'un des nouveaux modules disponibles uniquement en LGPL 3, vous devrez vous plier aux exigences de la nouvelle version.

Impact sur KDE

Lors du lancement du projet KDE, un environnement de bureau libre entièrement basé sur Qt, un accord juridique a été établi entre Trolltech (qui développait Qt) et KDE e.V. (l'entité légale derrière KDE), afin de s'assurer que Qt reste toujours disponible sous une licence libre : l'idée est que, si le projet quittait la scène du libre, le code puisse toujours évoluer, sans remettre en question KDE (cette utilisation d'une bibliothèque pas entièrement libre a été un frein à son adoption à la fin des années 1990). Cet accord est géré par la KDE Free Qt Foundation (lien 166), administrée tant par KDE e.V. que par Digia.

Ce changement de licence est une occasion d'améliorer le contrat liant ces deux entités. Notamment, toutes les plateformes, mobiles ou non, y sont intégrées (soit Windows, OS X, iOS, Windows RT, actuellement). Également, la fondation reçoit les droits sur toutes les contributions au Qt Project.

*Commentez la news de **Thibaut Cuvelier** en ligne : lien 167*

Sortie de Qt 5.4 Beta

Toute la puissance du Web avec Qt WebEngine

HTML5 est de nos jours une technologie importante et l'écosystème Qt se doit de proposer le meilleur support qui soit. Le Qt WebEngine, un projet de recherche et développement s'appuyant sur Chromium, a maintenant atteint une maturité certaine, en gérant totalement le bureau et les plateformes mobiles.

Qt WebEngine fournit une API commode, à la fois pour Qt Widget et pour Qt Quick, en utilisant le moteur Web du projet Chromium. Qt WebEngine ne se contente pas d'afficher du contenu Web, il tire aussi parti de l'ensemble de la pile graphique de Qt, ce qui vous permet de mélanger une surcouche d'apparence native reposant sur les contrôles Qt Quick avec du contenu Web et des shaders OpenGL. Bien évidemment, le fait que Qt fournisse un ensemble d'outils pour la création d'applications fait de Qt WebEngine l'API de référence pour les moteurs Web.

Le nouveau module **Qt WebChannel** joue le rôle de pont entre QML/C++ d'un côté et HTML/-JavaScript de l'autre, permettant d'exposer des objets dérivant de QObject dans un contexte Web.

Pour les plateformes qui ne permettent pas la distribution de Qt WebEngine, mais aussi pour celles qui n'ont pas besoin d'un moteur Web aussi développé, Qt 5.4 introduit un nouveau module encore au stade de préversion : Qt WebView. Il intègre des contenus de type Web au sein d'une interface graphique native et est dès à présent disponible pour Android et iOS. Cela fait aussi de Qt WebView une solution pratique et légère pour afficher de la documentation au format HTML au sein d'une application. Qt WebView sera disponible en même temps que la version finale de Qt 5.4 en qualité de supplément.

Qt 5.4 contient aussi le module Qt WebKit. Ce dernier est toujours supporté, mais il est considéré comme terminé et ne recevra donc plus aucune nouvelle fonctionnalité. Par ailleurs, ce module reposant sur WebKit sera déprécié dans une version future, étant donné que le nouveau module Qt WebEngine est équivalent au niveau des fonctionnalités. Notez par ailleurs que, dans la plupart des cas, le port de Qt WebKit à Qt WebEngine se déroule simplement et sans problème.

Gestion complète de Windows RT – publication dans le Windows Store

Le port de Qt pour Windows RT fut introduit dans Qt 5.3 Beta. Grâce aux retours des utilisa-

teurs, il a été grandement amélioré et de nombreuses API manquantes ont été implémentées. Qt 5.4, gère maintenant totalement la plateforme WinRT.

Avec Qt 5.4, la plupart des modules du framework sont disponibles sur WinRT, par exemple Qt Quick, Qt Quick Controls, Multimedia, Positioning, Network (y compris SSL/TLS), Core et GUI. Avec Qt pour WinRT, pour pouvez maintenant déployer vos applications sur le Windows Store et ainsi viser les Windows Phone 8.1 (et plus récents) et les tablettes Windows avec l'interface Modern UI.

Nouvelles conditions de licence – introduction de la LGPLv3

Comme annoncé plus tôt (lien 168), la version libre de Qt 5.4 est aussi disponible sous licence LGPLv3. Cette nouvelle licence permet à « The Qt Company » de proposer des composants à haute valeur ajoutée à la communauté Qt sans que cela n'impacte les affaires de l'entité commerciale.

Améliorations pour Windows

En plus de WinRT, Qt 5.4 apporte de nombreuses améliorations aux utilisateurs de Windows. Qt 5.4 apporte le support des écrans High DPI. Quelle que soit la résolution de la dalle ou sa résolution, Qt 5.4 maintient un affichage cohérent. Cette nouvelle version mineure apporte le support du HighDPI pour Windows, ainsi qu'un certain nombre d'améliorations pour les autres plateformes telles qu'OS X et X11.

Le support du HighDPI est encore considéré comme étant expérimental dans Qt 5.4 et doit être activé à l'aide de variables d'environnement. Si vous souhaitez en savoir plus sur le support du HighDPI, vous pouvez jeter un coup d'œil à la documentation : lien 169.

Qt 5.4 apporte la possibilité de choisir dynamiquement pendant le lancement de l'application s'il faut utiliser ANGLE ou OpenGL sous Windows. Il est donc possible d'utiliser au choix opengl32.dll ou bien l'implémentation OpenGL ES 2.0 d'ANGLE dans une application Qt sans avoir besoin de fournir deux binaires différents. Cela simplifie grandement la création d'applications Qt Quick pour les machines Windows. La sélection dynamique de l'implémentation d'OpenGL n'est cependant pas activée par défaut dans les binaires de Qt 5.4 Beta.

En plus de cette nouvelle fonctionnalité, cette nouvelle version apporte un nombre significatif de corrections de bogues et d'améliorations quant au support de Windows.

Autres améliorations du côté graphique

Du côté graphique, le support du HighDPI et le changement dynamique d'OpenGL ne sont pas les seules nouveautés concernant Windows. Une des plus importantes nouveautés est la classe QOpenGLWidget, le remplaçant moderne du vieillissant QGLWidget de Qt 4. QOpenGLWidget est un nouveau widget permettant d'afficher du contenu rendu par OpenGL et qui peut être utilisé comme n'importe quel autre widget. Cela permet de déprécier l'ancien module Qt OpenGL, étant donné que toutes ses fonctionnalités et bien d'autres encore sont maintenant assurées par d'autres modules.

Qt 5.4 amène aussi une nouvelle API, nommée QQuickRenderControl, qui permet un rendu efficace de scènes Qt Quick 2 dans des tampons de scènes. Le contenu peut alors être utilisé aussi bien dans des applications basées sur Qt ou bien sur des générateurs de rendu tiers. Avec Qt 5.4, il est maintenant possible d'adopter et d'emballer des contextes OpenGL existants en tant que QOpenGLContext. Cela facilite l'utilisation de contenu généré avec Qt dans d'autres moteurs de rendu. En plus d'API graphiques additionnelles, Qt 5.4 apporte deux nouvelles classes, QOpenGLWindow et QRasterWindow et introduit le support d'images avec 10 bits par canal.

QOpenGLContext est maintenant capable d'adopter des contextes natifs existants (EGL, GLX...). Cela permet une interopérabilité entre Qt et d'autres briques logicielles, comme des moteurs de jeux. Pour plus d'informations à ce propos, vous pouvez consulter l'introduction à QOpenGLWidget sur le blog du Qt Project : lien 170.

Bluetooth basse consommation

Qt 5.4 présente une implémentation du Bluetooth basse consommation, qui permet la communication avec un nombre important de capteurs intelligents comme les habits. Cette implémentation ne fonctionne, pour le moment, qu'avec BlueZ 4 et 5 sur Linux uniquement ; la gestion d'autres plateformes telles qu'iOS et Android sera ajoutée dans les prochaines versions de Qt. Avec Qt 5.4, seules les fonctionnalités centrales telles que définies par les spécifications « Bluetooth 4.0 » sont supportées, c'est-à-dire qu'il n'est possible que de créer des clients Bluetooth basse consommation.

Si vous êtes intéressés par le Bluetooth basse consommation, vous pouvez consulter la vue d'ensemble du module Qt Bluetooth : lien 171. Les retours des utilisateurs seront très importants pour permettre à ce module de dépasser le stade de préversion.

Démarrage plus rapide, déploiement plus léger et style natif pour Android

Les développeurs ont beaucoup travaillé à l'amélioration du support d'Android. Ainsi, cette version apporte un nombre considérable de nouvelles fonctionnalités pour Android. L'analyse des importations QML est maintenant gérée, ce qui aide à réduire la taille des paquets à déployer. Par ailleurs, le cache prégénéré des ressources introduit par Qt 5.3 permet de démarrer une application en moins de temps qu'il ne faut pour le dire.

Du côté de l'interface utilisateur, les widgets et les contrôles Qt Quick ont maintenant une appa-

rence native. Avec les précédentes versions de Qt, cela était déjà possible, mais il fallait utiliser Ministro. À partir de Qt 5.4, tous les widgets et contrôles créés avec Qt Quick comme les boutons, les curseurs, les barres de progression auront une apparence native par défaut.

Gestion améliorée d'iOS

Cette nouvelle version mineure de Qt améliore grandement le support d'iOS en amenant un nombre important de correctifs pour iOS 8 et Xcode 6. En plus de ces améliorations mineures et des habituelles corrections de bogues, cette version apporte différentes nouvelles fonctionnalités comme le modèle de sélection tactile de texte. Au lieu de la précédente approche type « bureau » (appuyer et glisser), on peut maintenant appuyer et maintenir appuyé pour sélectionner du texte. Qt affichera des poignées de sélection qui peuvent être tirées, ainsi qu'une note contextuelle, comme avec les applications natives. Le menu d'édition peut par ailleurs être modifié grâce au contrôle Qt Quick.

Plus généralement, l'apparence des applications Qt se rapproche de plus en plus des applications natives. Par ailleurs, beaucoup de correctifs améliorant la stabilité et les performances ont été apportés aux parties du code gérant le fenêtrage, l'écran, l'orientation et la géométrie, ainsi que la gestion du clavier virtuel.

Gestion d'OS X 10.10, de la signature de code et autres améliorations pour OS X

Qt 5.4 améliore significativement la gestion d'OS X 10.10 Yosemite. Les développeurs ont travaillé dur pour améliorer et corriger les problèmes de style et les autres erreurs rencontrées lors de l'utilisation conjointe d'OS X 10.10 et de Qt 5.4. Les applications créées avec une version antérieure de Qt fonctionneront avec cette nouvelle version d'OS X 10.10, mais il peut y avoir des erreurs de style, cela dépend de votre application. En plus d'OS X 10.10, beaucoup de bogues spécifiques à OS X remontés par les utilisateurs ont été corrigés et de nouveaux bogues seront corrigés avant la sortie de la version finale.

Une nouveauté significative de Qt 5.4 est le support de la signature de code, requise sur OS X 10.10 (et 10.9.5) pour publier des applications sur le Mac AppStore. Tout cela est encore en développement, mais la fonctionnalité initiale est déjà présente dans cette préversion. Sa version complète sera normalement proposée dans la version finale de Qt 5.4.

Tablettes Wacom, Wayland, mise à jour et de Qt Creator et améliorations diverses

Le support des tablettes Wacom a été amélioré et unifié au sein des différentes plateformes dans Qt 5.4, grâce à l'aide et au support des développeurs KDE de Krita. QTabletEvent contient maintenant les informations concernant les boutons du stylet, la rotation et la pression tangentielle de ce dernier sont maintenant des valeurs unifiées vis-à-vis des différentes plateformes. Par ailleurs, l'événement de proximité contient maintenant le type de stylet et l'outil en cours d'utilisation. Ces différentes améliorations ne feront plus de la gestion des tablettes un obstacle à la migration vers Qt 5.

Qt 5.4 contient maintenant le module Qt Wayland. Cela vous permet d'exécuter vos applications sur Weston, le compositeur de référence pour Wayland. La compatibilité avec Weston et de Wayland en est encore à ses premiers pas, Qt ne fournit pas encore l'expérience d'environnement de bureau comme vous pouvez la retrouver sur Windows, Cocoa ou xcb. Quoi qu'il en soit, Weston et Wayland fournissent dès à présent un système de fenêtrage qui peut se révéler utile dans le monde de l'embarqué. Le module Qt Wayland n'est livré que sous la forme de code source. Le support de Wayland dans Qt est un objectif important, notamment concernant la création de périphériques.

Qt 5.4 Beta fournit une mise à jour de l'environnement de référence, Qt Creator 3.2.2, ce qui apporte différentes améliorations, notamment pour les exemples concernant Xcode 6 et l'utilisation du simulateur iOS.

Conclusion

Cet article présente les nouveautés les plus significatives de Qt 5.4, mais il a forcément fallu faire un choix. Pour la liste complète des nouveautés introduites, vous êtes vivement invité à consulter la page du wiki listant les nouveautés de cette nouvelle version mineure : lien 172. Vous pouvez aussi télécharger Qt 5.4 Beta et vous faire votre propre opinion.

Qt 5.4 Beta est dès à présent disponible sur le site Web du Qt Project : lien 173. Merci de télécharger cette nouvelle version et de faire remonter les bogues via la liste de diffusion ou l'application de gestion de bogues : lien 174.

*Commentez la news d'**Arnold Dumas** en ligne : lien 175*

Liste des liens

Page 7
lien 1 : ... http://houri-mohamed.developpez.com/tutoriels/oracle/optimisations-ldd/

Page 8
lien 2 : ... http://www.developpez.com/actu/69155/
lien 3 : ... https://netbeans.org/downloads/
lien 4 : ... http://www.developpez.net/forums/d1425729/java/edi-outils-java/netbeans/netbeans-s-aligne-java-8-a/#post7967835

Page 9
lien 5 : ... http://mail.openjdk.java.net/pipermail/announce/2014-September/000182.html
lien 6 : ... https://bugs.openjdk.java.net/browse/JDK-8043364
lien 7 : ... http://www.developpez.net/forums/d1467788/java/general-java/java-pourrait-etre-dote-d-interpreteur-boucle-lecture-evaluation-impression/

Page 10
lien 8 : ... http://disqus.com/gavinking/
lien 9 : ... http://lmauzaize.developpez.com/tutoriels/ceylon/presentation/
lien 10 : ... http://lmauzaize.developpez.com/tutoriels/ceylon/bases/
lien 11 : ... http://lmauzaize.developpez.com/tutoriels/ceylon/typage/
lien 12 : ... http://modules.ceylon-lang.org/repo/1/ceylon/language/1.0.0/module-doc/Null.type.html
lien 13 : ... http://modules.ceylon-lang.org/repo/1/ceylon/language/1.0.0/module-doc/null.object.html

Page 13
lien 14 : ... http://docs.oracle.com/javase/7/docs/api/java/lang/StringBuilder.html
lien 15 : ... http://docs.oracle.com/javase/7/docs/api/java/lang/StringBuffer.html
lien 16 : ... http://docs.oracle.com/javase/7/docs/api/java/lang/Object.html
lien 17 : ... http://docs.oracle.com/javase/7/docs/api/java/lang/CharSequence.html
lien 18 : ... http://docs.oracle.com/javase/7/docs/api/java/lang/Appendable.html

Page 15
lien 19 : ... http://modules.ceylon-lang.org/repo/1/ceylon/language/1.0.0/module-doc/Object.type.html
lien 20 : ... http://modules.ceylon-lang.org/repo/1/ceylon/language/1.0.0/module-doc/Anything.type.html
lien 21 : ... http://docs.oracle.com/javase/7/docs/api/java/util/List.html

Page 17
lien 22 : ... http://modules.ceylon-lang.org/repo/1/ceylon/language/1.0.0/module-doc/Object.type.html

Page 19
lien 23 : ... https://github.com/maulogan/ceylon-articles/tree/officiel

Page 20
lien 24 : ... http://lmauzaize.developpez.com/tutoriels/ceylon/typage/

Page 21
lien 25 : ... http://www.7-zip.org/sdk.html

Page 23
lien 26 : ... http://www.developpez.com/redirect/1275
lien 27 : ... http://intel.developpez.com/tutoriels/android/decouverte-sdk-compression-bibliotheque-native-application/

Page 24
lien 28 : ... http://jeux.developpez.com/tutoriels/OpenGL-moderne/tutoriel-9-indexation-VBO/
lien 29 : ... http://jeux.developpez.com/tutoriels/OpenGL-moderne/outils-et-liens-utiles/
lien 30 : ... http://jeux.developpez.com/tutoriels/OpenGL-moderne/compteur-FPS/

Page 25

lien 31 : ... http://jeux.developpez.com/evenements/we-jeux/2/

lien 32 : ... http://jeux.developpez.com/evenements/we-jeux/3/

lien 33 : ... http://chat.developpez.com/

lien 34 : ... http://www.developpez.net/forums/d1454009/applications/developpement-2d-3d-jeux/evenement-22-24-aout-quatrieme-week-end-programmation-jeux-video-developpez-com/

lien 35 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/pack-we4.zip

lien 36 : ... http://www.developpez.net/forums/d1465460/applications/developpement-2d-3d-jeux/projets/we-jv4-projet-bloc-gnop/

lien 37 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Fusoy/Bloc_Gnop_Windows.zip

lien 38 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Fusoy/Bloc_Gnop_Linux.zip

lien 39 : ... http://www.developpez.net/forums/d1465581/applications/developpement-2d-3d-jeux/projets/we-jv4-tkboulderdash-python3-tkinter-port-of-the-famous-game/

lien 40 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/tarball69/tkBoulderDash-v0.5.zip

lien 41 : ... http://www.developpez.net/forums/d1465459/applications/developpement-2d-3d-jeux/projets/we-jv4-repaint-precis-chanceux/

lien 42 : ... http://unlicense.developpez.com/

Page 26

lien 43 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/eclesia/Repaint.zip

lien 44 : ... https://bitbucket.org/Eclesia/unlicense-gamebase

lien 45 : ... http://www.developpez.net/forums/d1465464/applications/developpement-2d-3d-jeux/projets/projet-faillite-we-jv4-grill-weekend-tetrisnet/

lien 46 : ... http://www.developpez.net/forums/d1465676/applications/developpement-2d-3d-jeux/projets/we-jv4-the-adventures-of-link/

lien 47 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Light99/%5BWE-JV4%5D%20The%20Adventures%20of%20Link.zip

lien 48 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/online/Light99/

lien 49 : ... http://www.developpez.net/forums/d1465602/applications/developpement-2d-3d-jeux/projets/we-jv4-cara-cara-carambar/

lien 50 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Guntha/WEJV4_Guntha_version4_Windows.zip

lien 51 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Guntha/WEJV4Guntha-debug_version4.apk

lien 52 : ... http://www.developpez.net/forums/d1465663/applications/developpement-2d-3d-jeux/projets/we-jv4-2d-plate-forme-construct2-opus-magnum/

lien 53 : ... http://syllab.fr/projets/games/opusmagnum/

lien 54 : ... http://www.developpez.net/forums/d1465539/applications/developpement-2d-3d-jeux/projets/we-jv4-p-sharpld30-wcom-vb-net-p-dx11/

Page 27

lien 55 : ... http://www.youtube.com/v/im_cpqLqREo&autoplay=1

lien 56 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/ShadowTzu/WCOM.zip

lien 57 : ... http://www.developpez.net/forums/d1465679/applications/developpement-2d-3d-jeux/projets/we-jv4-caitland-text-adventure/

lien 58 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/online/Meiryo/caitland.min.html

lien 59 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Meiryo/caitland-min.rar

lien 60 : ... http://www.developpez.net/forums/d1465498/applications/developpement-2d-3d-jeux/projets/we-jv4-nullotron-presentation-jeu/

lien 61 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/CodeurPlusPlus/NULLOTRON.zip

lien 62 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/CodeurPlusPlus/HARDNULLOTRON.zip
lien 63 : ... http://www.developpez.net/forums/d1465478/applications/developpement-2d-3d-jeux/projets/we-jv4-super-mandala-omega/
lien 64 : ... http://www.developpez.net/forums/d1465519/applications/developpement-2d-3d-jeux/projets/we-jv4-stylo-presentation-jeu/
lien 65 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/Pikel/Le%20Stylo.zip
lien 66 : ... http://www.developpez.net/forums/d1465653/applications/developpement-2d-3d-jeux/projets/we-jv4-spacedefend-td/
lien 67 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/yetimothee/spaceDefend_24082014_22h00.zip
lien 68 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/yetimothee/spaceDefend_src_24082014_22h00.zip

Page 28

lien 69 : ... http://www.developpez.net/forums/d1465457/applications/developpement-2d-3d-jeux/projets/we-jv4-mix-weekend-tetriscommand/
lien 70 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/LittleWhite/WE-JV4-TetrisCommand-windows.zip
lien 71 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/LittleWhite/WE-JV4-TetrisCommand-source-2014-08-24-2042.tar.bz2
lien 72 : ... http://www.developpez.net/forums/d1465624/applications/developpement-2d-3d-jeux/projets/projet-cours-we-jv4-mini-supertux/
lien 73 : ... http://www.developpez.net/forums/d1465487/applications/developpement-2d-3d-jeux/projets/we-jv4-papi-commando-online/
lien 74 : ... http://jeux.developpez.com/evenements/we-jeux/4/jeux/fichiers/MoDDiB/PapiOnline4.zip
lien 75 : ... http://www.developpez.net/forums/d1465661/applications/developpement-2d-3d-jeux/projets/we-jv4-blackjack-ligne/
lien 76 : ... http://nateriver883.magix.net/
lien 77 : ... http://www.developpez.net/forums/d1465473/applications/developpement-2d-3d-jeux/projets/we-jv4-hexagoune-epilad-epitouille/
lien 78 : ... http://www.developpez.net/forums/d1465677/applications/developpement-2d-3d-jeux/projets/we-jv4-galaxy-bird/
lien 79 : ... http://jeux.developpez.com/evenements/we-jeux/4/

Page 29

lien 80 : ... http://jqueryui.com
lien 81 : ... http://yuilibrary.com/
lien 82 : ... http://dojotoolkit.org/reference-guide/1.10/dojox/widget.html
lien 83 : ... https://www.polymer-project.org/
lien 84 : ... http://mozbrick.github.io/
lien 85 : ... http://www.polymer-project.org/articles/polymer-xtag-vanilla.html

Page 30

lien 86 : ... http://addyosmani.github.io/fitc-wccdt/
lien 87 : ... http://www.html5rocks.com/en/tutorials/webcomponents/template/

Page 31

lien 88 : ... http://validator.w3.org/
lien 89 : ... http://www.html5rocks.com/en/tutorials/webcomponents/shadowdom/#toc-separation
lien 90 : ... http://webcomponents.org/articles/web-components-best-practices/
lien 91 : ... http://webcomponents.org/

Page 32

lien 92 : ... http://www.polymer-project.org/articles/accessible-web-components.html

Page 33

lien 93 : ... http://tjvantoll.com/2012/06/30/creating-a-native-html5-datepicker-with-a-fallback-to-jquery-ui/

lien 94 : ... https://github.com/x-tag/datepicker

Page 34

lien 95 : ... http://www.html5rocks.com/en/tutorials/webcomponents/imports/

Page 35

lien 96 : ... http://sylvainpv.developpez.com/tutoriels/javascript/guide-templating-client/

Page 36

lien 97 : ... https://github.com/zenorocha/facebook-button
lien 98 : ... https://github.com/cesarwbr/google-analytics-element
lien 99 : ... https://github.com/faunt/domrgn
lien 100 : ... http://component.kitchen/components/google-map
lien 101 : ... http://garstasio.github.io/file-input/components/file-input/
lien 102 : ... http://bosonic.github.io/elements.html#b-datepicker
lien 103 : ... http://www.polymer-project.org/docs/elements/core-elements.html#core-ajax
lien 104 : ... https://github.com/passy/x-pokemon
lien 105 : ... https://github.com/beldar/boris-bikes

Page 37

lien 106 : ... http://component.kitchen/
lien 107 : ... http://customelements.io/
lien 108 : ... http://sylvainpv.developpez.com/publications/web-components-debat/

Page 38

lien 109 : ... http://www.debian.org/
lien 110 : ... http://www.inetdoc.net/pdf/intro.analyse.pdf
lien 111 : ... https://www.wireshark.org/
lien 112 : ... http://wiki.wireshark.org/ProtocolReference
lien 113 : ... http://wiki.wireshark.org/CaptureSetup/NetworkMedia
lien 114 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.menus

Page 39

lien 115 : ... https://sites.google.com/site/fullycapable/

Page 40

lien 116 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.capture.html
lien 117 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.framelist

Page 41

lien 118 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.filter.html

Page 42

lien 119 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.postfilter
lien 120 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.postfilter

Page 43

lien 121 : ... https://www.wireshark.org/docs/dfref/

Page 47

lien 122 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.protocollist
lien 123 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.hexa
lien 124 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.protocollist

Page 48

lien 125 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.gui.html#wireshark.hexa

Page 49

lien 126 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.lab.icmp.html#traceroute.ttl-exceeded
lien 127 : ... http://www.inetdoc.net/travaux_pratiques/intro.analyse/wireshark.lab.icmp.html#traceroute.udp
lien 128 : ... https://www.wireshark.org/download/docs/user-guide-a4.pdf
lien 129 : ... http://www.sans.org/security-resources/tcpip.pdf
lien 130 : ... http://www.inetdoc.net/travaux_pratiques/config.interface.lan/
lien 131 : ... http://media.techtarget.com/searchEnterpriseLinux/downloads/284_EPS_04.pdf
lien 132 : ... http://gaia.cs.umass.edu/ethereal-labs/
lien 133 : ... http://inetdoc.developpez.com/tutoriels/analyse-reseau-wireshark/

Page 50
lien 134 : ... http://www.developpez.net/forums/d1378843/webmasters-developpement-web/javascript-ajax-typescript-dart/typescript/introduction-langage-typescript/
lien 135 : ... http://yahiko.co.nf/RNGVisualizer/
lien 136 : ... http://www.univ-irem.fr/reperes/articles/65_article_447.pdf
lien 137 : ... http://www2.lbl.gov/Science-Articles/Archive/pi-random.html

Page 51
lien 138 : ... http://fr.wikipedia.org/wiki/Générateur_congruentiel_linéaire
lien 139 : ... http://fr.wikipedia.org/wiki/RANDU

Page 53
lien 140 : ... http://www.honeylocust.com/javascript/randomizer.html

Page 54
lien 141 : ... http://en.wikipedia.org/wiki/Xorshift
lien 142 : ... http://xorshift.di.unimi.it/
lien 143 : ... http://fr.wikipedia.org/wiki/Mersenne_Twister

Page 55
lien 144 : ... http://www.developpez.net/forums/d1470523/applications/developpement-2d-3d-jeux/generation-aleatoire-d-univers/
lien 145 : ... http://yahiko.co.nf/universeGenerator/

Page 56
lien 146 : ... http://fr.wikipedia.org/wiki/Théorème_central_limite
lien 147 : ... http://fr.wikipedia.org/wiki/Méthode_de_Box-Muller
lien 148 : ... http://ram-0000.developpez.com/tutoriels/cryptographie/
lien 149 : ... https://qrng.physik.hu-berlin.de/
lien 150 : ... http://www.random.org/
lien 151 : ... https://github.com/yahiko00/Random-Number-Generator-Visualizer
lien 152 : ... http://yahiko.developpez.com/tutoriels/generateur-nombre-aleatoire/

Page 61
lien 153 : ... http://openoffice-libreoffice.developpez.com/tutoriels/openoffice-libreoffice/calc-tableur-mysql/

Page 66
lien 154 : ... http://sql.developpez.com/

Page 67
lien 155 : ... http://vviale.developpez.com/tutoriels/openoffice-libreoffice/base-table-vue-requete/

Page 78
lien 156 : ... http://vviale.developpez.com/tutoriels/openoffice-libreoffice/base-formulaire/

Page 79
lien 157 : ... https://qt.gitorious.org/qt-creator/qt-creator/source/3.2:dist/changes-3.2.0
lien 158 : ... https://download.qt-project.org/official_releases/qtcreator/3.2/3.2.0/
lien 159 : ... https://bugreports.qt-project.org/
lien 160 : ... http://lists.qt-project.org/mailman/listinfo/qt-creator
lien 161 : ... http://www.developpez.net/forums/d1462855/c-cpp/bibliotheques/qt/edi/qt-creator/sortie-qt-creator-3-2-0-a/#post7933978

lien 162 : ... http://qt.developpez.com/actu/3365/Qt-4-5-sous-licence-LGPL/

Page 80

lien 163 : ... http://qt.developpez.com/actu/66944/Qt-WebEngine-le-nouveau-moteur-Web-de-Qt-apercu-technique-du-port-de-Chromium-pour-Qt/

lien 164 : ... https://qt.gitorious.org/qt/qtcanvas3d

lien 165 : ... https://qt.gitorious.org/qt/qtwebview

lien 166 : ... http://kde.org/community/whatiskde/kdefreeqtfoundation.php

lien 167 : ... http://www.developpez.net/forums/d1464911/c-cpp/bibliotheques/qt/nouvelle-licence-qt-lpgl-3-a/

Page 81

lien 168 : ... http://qt.developpez.com/actu/74292/Nouvelle-licence-pour-Qt-le-framework-sera-egalement-disponible-sous-la-LGPL-3-avec-de-nouveaux-modules-libres/

lien 169 : ... http://doc-snapshot.qt-project.org/qt5-5.4/highdpi.html

lien 170 : ... http://blog.qt.digia.com/blog/2014/09/10/qt-weekly-19-qopenglwidget/

lien 171 : ... http://doc-snapshot.qt-project.org/qt5-5.4/qtbluetooth-le-overview.html

Page 82

lien 172 : ... http://qt-project.org/wiki/New-Features-in-Qt-5.4

lien 173 : ... http://download.qt-project.org/development_releases/qt/5.4/

lien 174 : ... http://bugreports.qt-project.org/

lien 175 : ... http://www.developpez.net/forums/d1468332/c-cpp/bibliotheques/qt/sortie-qt-5-4-beta/#post8002434