

Ciboulette Balistique

Document de standards

James Jeffrey

Janie Brière

Maxime Damour

Charles Brouard

DÉVELOPPEMENT DE LOGICIELS D'ENTREPRISES

Table des matières

[Nomenclature 1](#_Toc477412458)

[Projets et solutions 1](#_Toc477412459)

[Formes et scripts SQL 1](#_Toc477412460)

[Boutons, les objets et objets hérités 2](#_Toc477412461)

[Objets dans les formes personnelles 2](#_Toc477412462)

[Les procédures et autres variables 3](#_Toc477412463)

[Communication avec l’usager 3](#_Toc477412464)

# Nomenclature

## Projets et solutions

Les taches à faire sont séparées ainsi :

* James Jeffrey : gestion des réservations
* Janie Brière : gestion des chambres
* Maxime Damour : gestion des arrivées et des départs
* Charles Brouard : gestion des transactions

Chaque projet individuel suit la forme Prj\_Module\_Nom. Tous seront dans un dossier de groupe nommé Prj\_SystemeCB.

Le dossier « Prj\_menu » contenient le menu ainsi que la librairie graphique et tous les boutons. Ainsi, tous les projets individuels vont hériter de la Form1 de la librairie nommée « Prj\_lib\_graphique ».

Les ressources graphiques sont stockées à la racine du projet système dans le dossier images.

## Formes et scripts SQL

Les formes des projets individuels porteront le nom de la tâche du programmeur (ex. Chambre.cs). Le nom des autres formes, par exemple pour les listes de sélection, sera à la discrétion du programmeur tant que le nom est significatif.

Il y a deux scripts SQL l’un permet de créer la base de données et l’autre de la remplir, plutôt que de créer plusieurs fichiers d’insertion nous avons tout mis dans le même fichier SQL séparé par des balises. Exemple :

/\*\*\*\*\*\*\*\*\*\*

\*

\* INSERT CLIENT

\*

\*\*\*\*\*\*\*\*\*\*\*/

|  |  |
| --- | --- |
| CreationDBCB.sql | Pour la création de la base de données (tables, liaisons, etc) |
| Insert\_Base | Pour les insertions dans la base de données |

Nous partirons de cette base pour chaque présentation.

Les noms des tables, des champs et des clés primaires qui sont donnés dans le devis sont utilisés. Le nom des liaisons entre les tables par des clés secondaires est formé de la table qui a la clé secondaire et la table qui a cette clé en clé primaire. Exemple :

|  |  |
| --- | --- |
| La table qui utilise la clé : **Réservation** | La table liée : **Client** |
| Le nom de la liaison : **FK\_RESCLI** | |

C’est la table Réservation qui a la clé primaire de la table Client, le nom est donc formé.

## Boutons, les objets et objets hérités

Dans le menu du projet, il y a les objets (boutton) pour accéder aux projets individuels, ainsi que des objets esthétiques.

* Btn\_Reservation
* Btn\_Arrive
* Btn\_Chambre
* Btn\_Depart
* Btn\_Transaction
* Lab\_NomCie pour le nom de la compagnie
* PictureBox\_Logo pour le logo de la compagnie

Il y a ensuite les boutons dont chaque forme hérite, qui se trouve dans la librairie d’objets du Prj\_menu (Prj\_lib\_graphique) :

-objets hérités nommés :

|  |  |
| --- | --- |
| Nom | Description |
| btn\_cancel | Annuler les changements |
| btn\_next | Navigation : suivant |
| btn\_previous | Navigation : précédent |
| btn\_save | Sauvegarder (sur ajout ou modification) |
| btn\_function | Format de base pour les boutons |
| lab\_format | Format de base pour les labels |
| txt\_name | Format de base pour les textBox |
| btn\_delete | Entrer en mode suppression |
| btn\_edit | Entrer en mode édition |
| btn\_add | Entrer en mode ajout |
| Logo\_CB | Logo de la compagnie |
| Txt\_CB | Nom de la compagnie |

Dans le projet individuel, chacun héritera de la librairie d’objet mais possèdera un DataGridView contenant les données de la base de données.

Chaque DataGridView sera nommé : DGV\_(module). Exemple : DGV\_Chambre, DGV\_Reservation , DGV\_Client.

Pour ceux utilisés pour une ListView, le nom sera DGV\_Liste(module).

### Objets dans les formes personnelles

Les objets qui seront insérés dans les projets individuels sera de la forme TYPE \_FONCTION où :

- FONCTION = brève description sur leur nature

La plupart du temps, dans les formes, un label sera utilisé pour nommer une valeur, et un textBox sera utilisé pour afficher cette valeur. Ils auront ainsi la même fonction, mais pas le même type.

|  |  |
| --- | --- |
| Objet | Nom donné |
| button | btn\_( fonction) |
| label | lab\_( fonction) |
| panel | pan\_( fonction) |
| textBox | txtBox\_( fonction) |
| dateTimePicker | date\_( fonction) |

Dans le cas où il y a deux labels qui doivent avoir le même nom, le programmeur mettra, s’il y a lieu, le parent du label. La décision lui appartient, pour autant que les noms restent clairs.

Des standards seront aussi appliqués sur les outils pour la gestion des tables dans Visual Studio et ce, de la manière suivante :

|  |  |
| --- | --- |
| Objet | Nom donné |
| tableAdapter | TA\_(table) |
| bindingSource | BS\_(table) |
| dataGridView | DGV\_(module) |
| dataSet | DataSet\_(module) |

## Les procédures et autres variables

Chacun des projets aura des fonctions/procédure. La fonction qui remplira les tableAdapters sera nommée « fill() » et les fonctions qui associent des champs de la forme à des valeurs de la table seront nommées lien\_(table). Exemple : lien\_Reservation().

# Communication avec l’usager

La police d’écriture se fera en California FB, principalement de taille 11,25pt.

Les commandes qui ne sont pas encore implémentées feront apparaitre des messages box avec un message d’erreur. Le contenu du message est laissé au choix du programmeur.

Les objets qui sont cliquables seront indiquer par le curseur, qui se changera en main.

Lors du déroulement du logiciel, le programmeur pourra mettre, au besoin, un indicatif visuel (couleur, message box) pour porter l’attention de l’usager sur un manque de données ou un soucis dans l’utilisation du formulaire.