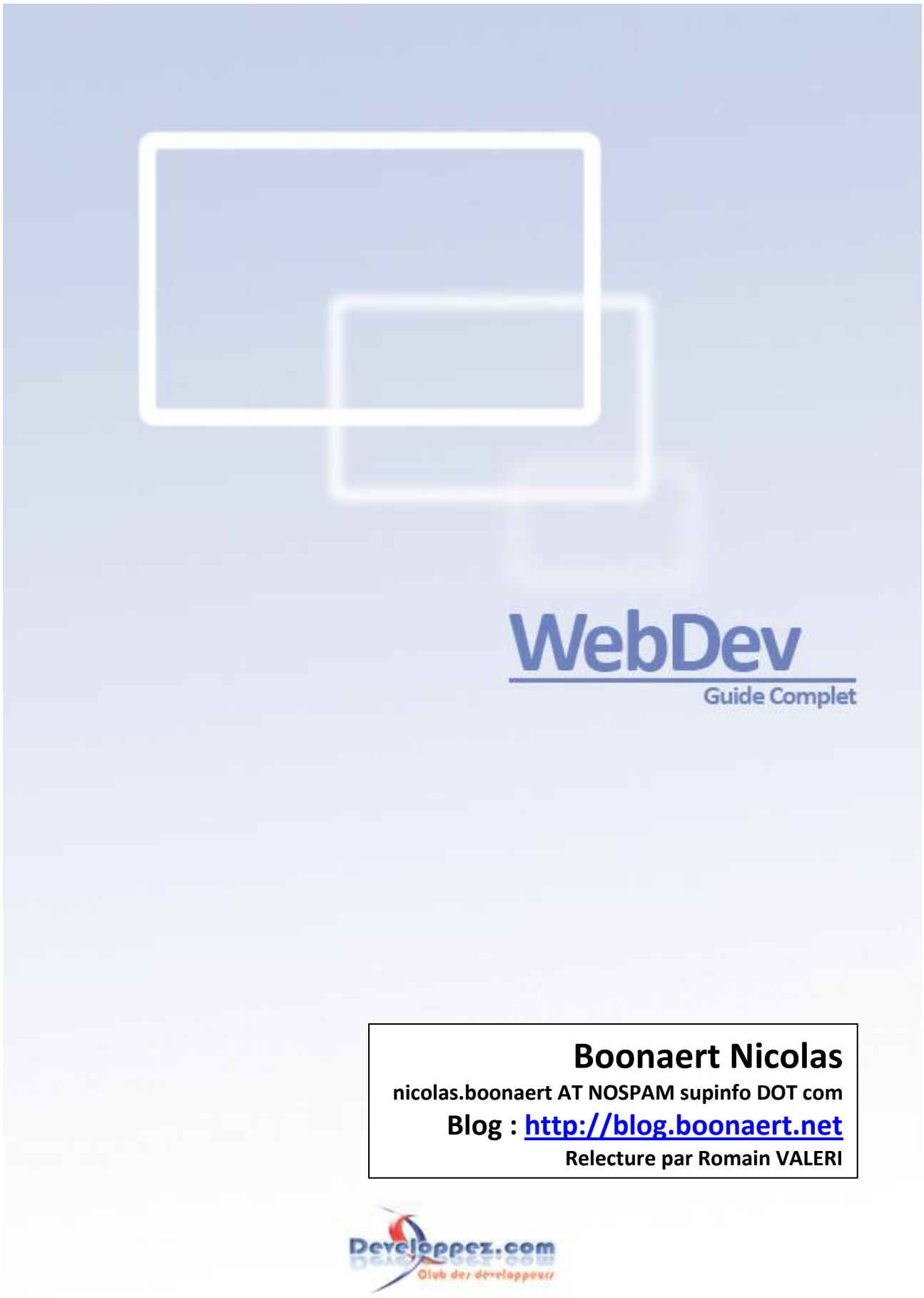


Boonaert Nicolas
nicolas.boonaert AT NOSPAM supinfo DOT com
Blog : http://blog.boonaert.net
Relecture par Romain VALERI

SOMMAIRE

1. Présentation générale

1.1. Présentation de l'article

Cet article traite principalement de la création d'un site internet dynamique à l'aide de Webdev.
Les technologies mises en œuvre ne feront intervenir que des pages dynamiques Webdev afin de ne pas traiter de cas spécifiques liés à d'autres types de page.

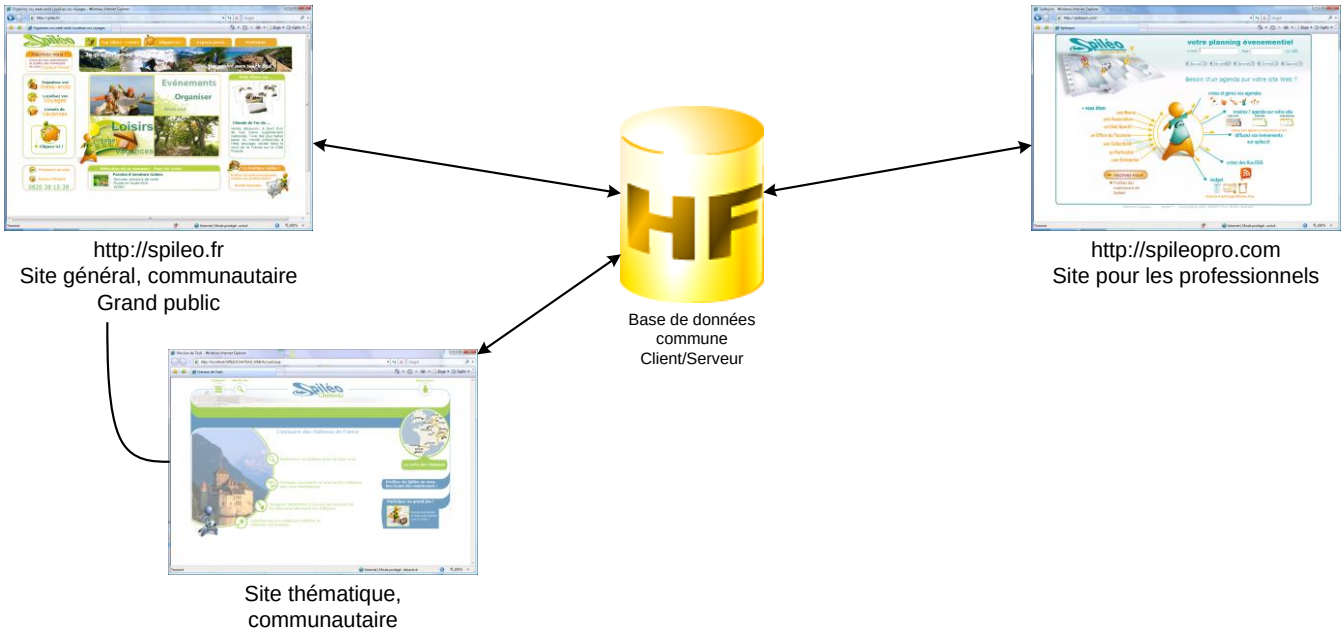
Cet article couvre également les différentes étapes de création de chacun des éléments ainsi que leur exploitation, avec une analyse des possibilités offertes par Webdev tout en présentant leurs avantages et leurs inconvénients.

Enfin dans cet article figurent de nombreuses astuces au sein de cet éditeur.
Mais tout d'abord il faut découvrir le projet sur lequel j'ai eu à travailler pendant la période de stage pour montrer un cas concret de réalisation et l'environnement du projet de développement, cependant certains éléments du projet initial ne peuvent pas être diffusés dans cet article pour des raisons de confidentialité c'est pourquoi j'ai fait le choix d'utiliser des exemples simplifiés et accessibles, ce qui améliore également la compréhension générale.

Cet article n'aborde pas le développement PHP à l'aide de Webdev, l'utilisation du RAD ou RID intégrés à Webdev.

1.1.1. Présentation du projet Spiléo

Le projet Spiléo est un projet de la société PC Logiciels.
Destiné à être utilisé sur Internet, c'est avant tout une plateforme regroupant plusieurs types d'utilisateurs. D'un côté le professionnel qui souhaite se référencer dans cet annuaire des événements et lieux touristiques, et de l'autre un particulier, visiteur du site général ou du site thématique profitant ainsi des données et participant à la vie de la communauté.



1.1.2. Présentation du projet de stage

Contexte de réalisation

Le projet de stage est un site dynamique qui repose sur une base de données que l'on a conçu lors du projet Spiléo.

Ce site est un site thématique qui doit pouvoir facilement être customisé pour changer de thème tant dans sa forme que dans le fond. Il était donc important de conserver cet impératif en tête lors de l'ensemble du développement.

Ce projet utilisant une plateforme commune à d'autres sites, il était important de travailler en équipe en se tenant informé tout au long du processus de développement et dans l'évolution de la plateforme.

Objectifs

Les objectifs étaient simples et complets, réaliser le site dans le temps imparti (soit les 3 mois de stage) en proposant une révision du travail déjà effectué. Ainsi réaliser une analyse de la plateforme Spiléo, et proposer des améliorations.

Rôle dans le projet

Ainsi, bien que la tâche principale qui m'était confiée était le développement du site thématique, j'ai dû participer à la conception et à l'amélioration de la plateforme commune aux autres sites.
Il m'était également demandé de fiabiliser et d'optimiser le site dans son ergonomie et dans l'interfaçage avec l'API Virtual Earth que l'on met en œuvre dans l'ensemble des sites tout en prenant en compte le besoin en sécurité de l'information. Ce point ne sera pas abordé pour cause de confidentialité.

1.2. Présentation de l'outil utilisé

Webdev peut être présenté comme un EDI (Environnement de Développement Intégré) même si certains préfère le terme AGL (Atelier de Génie Logiciel) pour ce logiciel.

Webdev est un environnement qui intègre de nombreuses fonctionnalités qui aident le développeur, le chef de projet, le décideur mais aussi l'ensemble des personnes qui participent au projet.

Il permet de réaliser des sites web sous plusieurs formes (qui seront décrites plus loin dans ce document) mais surtout il offre la possibilité de se concentrer sur les fonctionnalités offertes à l'utilisateur final, à la présentation des informations, en faisant abstraction de l'aspect purement technique.
Il reste cependant très important de maîtriser les technologies sous-jacentes notamment pour optimiser les traitements ou résoudre des problèmes qui peuvent survenir.

1.2.1. Cible et marché

Webdev ne se positionne pas véritablement face à d'autres gros éditeurs de l'industrie, autrement dit il ne faut clairement pas le voir face à des outils comme Visual Studio de Microsoft ou d'autres outils comme Eclipse, Netbeans, etc. …

L'outil existe maintenant depuis de nombreuses années et cible de préférence les TPE-PME qui souhaitent pouvoir répondre rapidement aux besoins de leurs clients.

L'outil, principalement en français, se décline en version anglaise également et annonce un nombre de licences de plus de 100 000 utilisateurs (Windev/Webdev/Windev Mobile réunis) avec une présence sur le marché français qui n'est pas sans marquer celui de l'emploi.

1.2.2. Possibilités

Webdev est principalement orienté pour des applications web, autrement dit des sites statiques, des sites dynamiques ou des composants réutilisables dans d'autres applications web.

Il intègre les dernières évolutions en terme de technologies internet avec pour exemple le plus mis en avant sur la toile, la technologie AJAX (Asynchronous JavaScript And XML) présente depuis la version 10 datant de 2006.

Toutes les technologies exploitées ou exploitables au sein de Webdev se veulent toujours relativement simples à mettre en œuvre. PC Soft garde cette volonté de donner accès à ces technologies sans être obligé de saisir l'intégralité de cette dernière.

Bien entendu, il est important et utile de connaître les mécanismes mis en œuvre pour véritablement saisir le fonctionnement complet, c'est d'ailleurs ce qui fait souvent la différence avec cet outil.

1.2.3. Langage utilisé

Le langage utilisé au sein de Webdev est principalement le W-Langage.
Ce langage étonne très souvent de par son aspect algorithmique puisqu'on peut utiliser des instructions ou fonctions directement en français (ou en anglais).

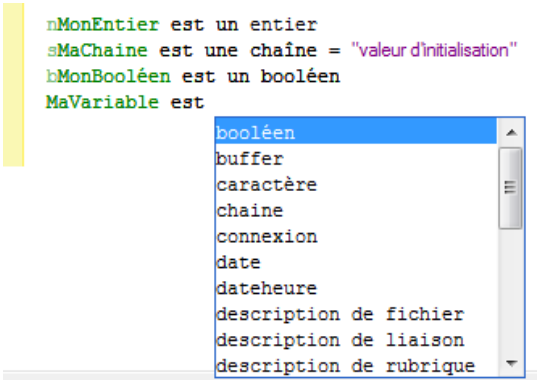
Ce W-Langage est utilisé pour le code coté serveur ou navigateur (la différence sera expliquée plus en détail dans la suite). Il permet bien entendu de développer orienté objet même si ce n'est pas l'utilisation première dans la société où le stage s'est déroulé.
Dans la suite du document, vous verrez pourquoi cette notion est abstraite dans le développement.

Voici quelques exemples de W-Langage

Déclaration de variables :

```
nMonEntier est un entier
sMaChaine est une chaîne = "valeur d'initialisation"
bMonBooléen est un booléen
```

Webdev, dans la charte de programmation par défaut, préfixe automatiquement les variables.
On peut également noter la présence de l'aide à la saisie dans l'ensemble du code.



Un exemple de code mettant en œuvre une boucle et une condition :

```
POUR i = 2 A 20
	SI (i modulo 2) <> 1 ALORS
		//Si c'est un nombre paire
		Info("Ma variable de boucle = " + i)
	FIN
FIN
```

Ce langage est directement compréhensible et ne nécessite que très rarement des commentaires, ce qui simplifie l'accessibilité et la maintenabilité du code.

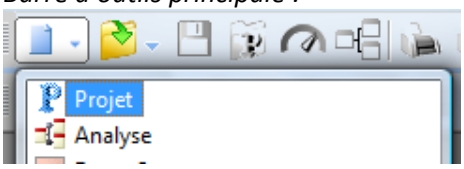
Du côté navigateur, on peut aussi utiliser du JavaScript au sein même de Webdev pour le code navigateur. Il est par ailleurs possible de faire interagir du code JavaScript à du code W-Langage et donc utiliser les librairies externes habituelles.

2. Réalisation du site Internet

2.1. Création du projet

Pour créer le projet, il est possible d'utiliser plusieurs méthodes :

Barre de menu : Fichier > Nouveau
Barre d'outils principale :



Le raccourci clavier : CTRL + N.

Dans tous les cas, on obtient la fenêtre suivante regroupant toutes les possibilités de création d'éléments. C'est à travers cette fenêtre que l'on pourra ensuite créer une nouvelle page, un nouvel état ou tout autre élément du projet.

Cette fenêtre regroupe les types d'éléments par couleur pour aider à rapidement saisir la nature de ces derniers, ainsi : Jaune : Projet ; Vert : Conception et Schéma (UML etc…) ; Bleu : Elément de projet ; Rouge : Analyse et base de données.

Les éléments en particuliers seront décrits dans leur contexte d'utilisation dans la suite du document.

2.1.1. Types de projets

A travers cette fenêtre on observe que l'on peut créer différents types de projets.
Le type de projet est à prendre en considération puisqu'il va imposer un certain nombre de restrictions.

Dynamique Webdev

Projet par défaut, il inclut l'ensemble des fonctionnalités et exploite parfaitement l'intégralité des composants. C'est le type de projet le plus complet et le plus fonctionnel, il peut intégrer des pages statiques, dynamiques ou semi-dynamiques qui s'appuieront au besoin sur une base de données.

Certains inconvénients sont tout de même présents, il est en effet important de les signaler : les sites dynamique Webdev doivent fonctionner sur un moteur de déploiement développé par PC Soft lors de la mise en production.
Ces sites dynamiques causent également quelques difficultés pour être indexés et référencés efficacement dans les moteurs de recherches.

Semi dynamique

Ce type de projet pourra contenir des pages semi-dynamiques et statiques. L'accès à la base de données sur ces pages sera donc plus limité (une seule table accessible par page).
Comme le projet dynamique, ce type de projet peut poser des problèmes pour le référencement et nécessite le moteur Webdev pour fonctionner en production.

Statique

Ce type de projet est utile pour des sites de présence Internet, il s'agit ni plus ni moins de pages HTML statiques. Ces pages n'ont donc pas de possibilités dynamiques (pas d'accès à une base de données), elles ne nécessitent donc pas de moteur de déploiement pour leur exploitation.

Dynamique AWP

Le mode AWP possède de nombreuses fonctionnalités du mode dynamique Webdev, certaines fonctionnalités importantes sont néanmoins manquantes en ce qui concerne le contexte serveur (session serveur) et certains traitements spécifiques.

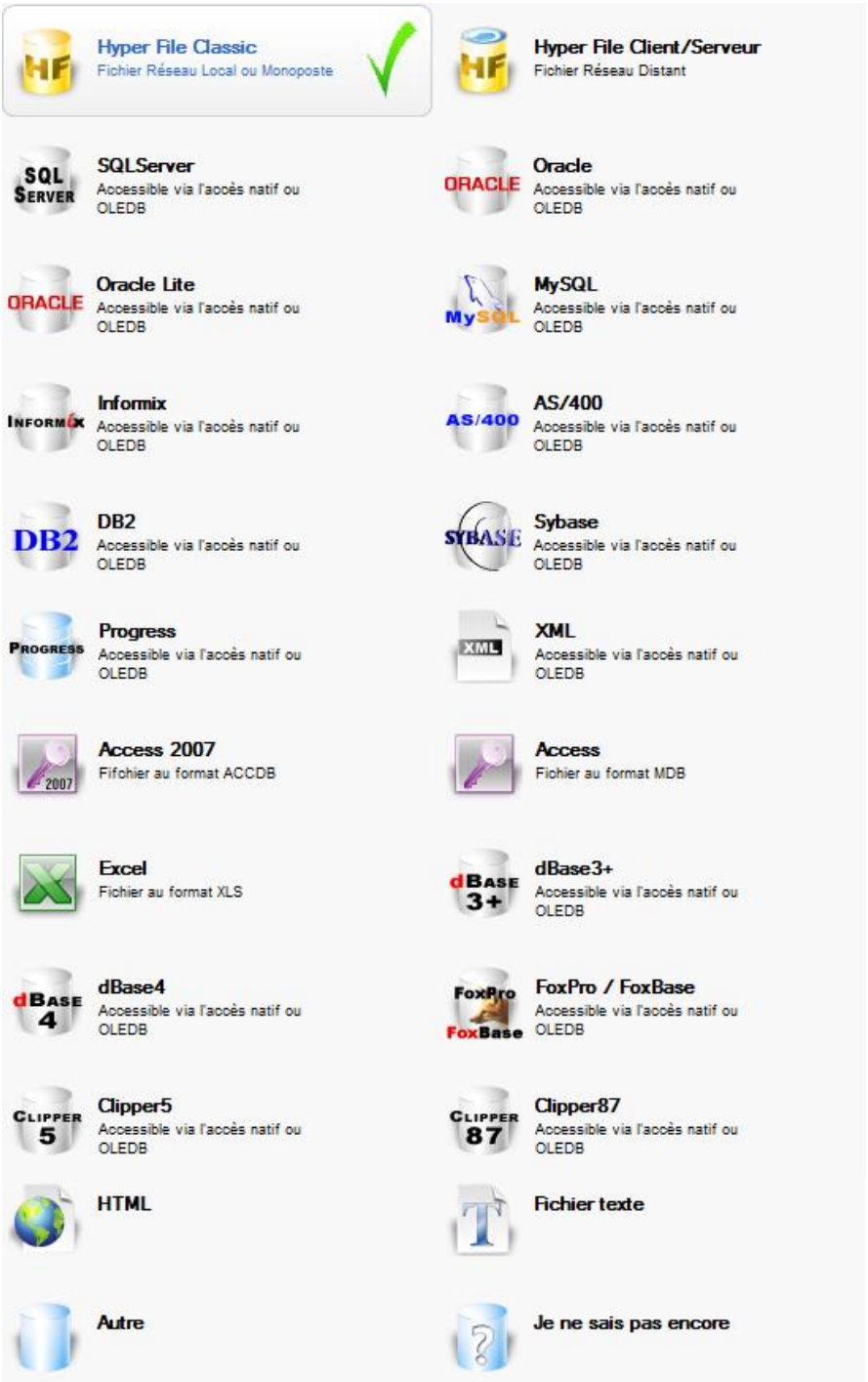
Dynamique PHP

Ce type de projet permet simplement de réaliser son site PHP au sein de Webdev, en utilisant la plupart des composants au sein de l'éditeur. Les technologies AJAX ne sont pas disponibles à l'heure actuelle (en version 11) dans ces pages PHP, il en va de même pour certains autres traitements qui demandent une exploitation particulière.
L'avantage de ce type de projet est que le résultat final se présente sous la forme de pages PHP dynamiques et hébergeables sur un serveur quelconque sans nécessiter de moteur de déploiement particulier.

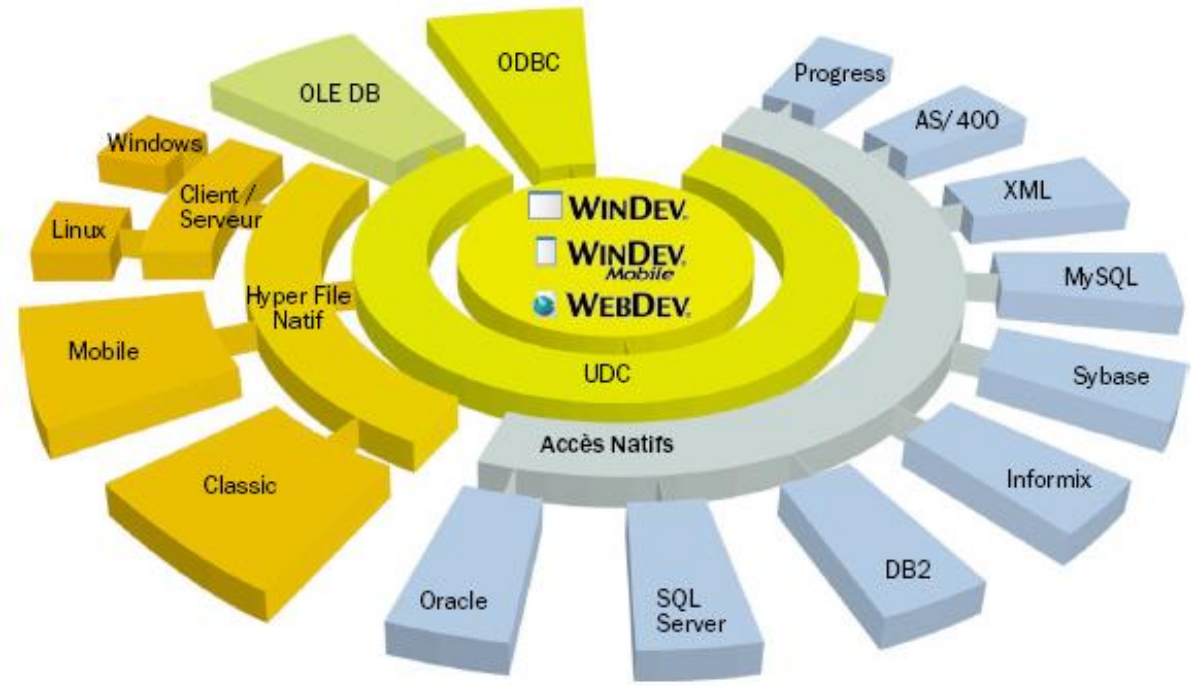
2.1.2. Base de données et accès natifs

Comme indiqué, on peut interfacer le projet avec plusieurs types de bases de données.
L'utilisation de certains de ces types est gratuite, c'est le cas des bases de données de type Hyper File.
Ce type de base de données est développé par PC Soft et possède les fonctionnalités habituelles d'un système de gestion de base de données relationnelle.

Le support des types de base de données plus répandues comme Oracle, MySQL ou bien d'autres, peut être ajouté en utilisant les accès natifs se présentant sous forme d'exécutables à installer et utilisables gratuitement ou non selon le type.



Ainsi, on peut remarquer que le projet créé peut s'interfacer avec n'importe quel type de base de données ce qui permet notamment d'intégrer des existants, l'ouverture est assuré par OLEDB et ODBC qui peuvent être utilisés dans la majorité des cas de figure.



UDC : Universal Data Connector (HLitRecherche, HExecuteRequête, Fichier, Rubrique, Liaison fichier automatique, SQLExec,…)

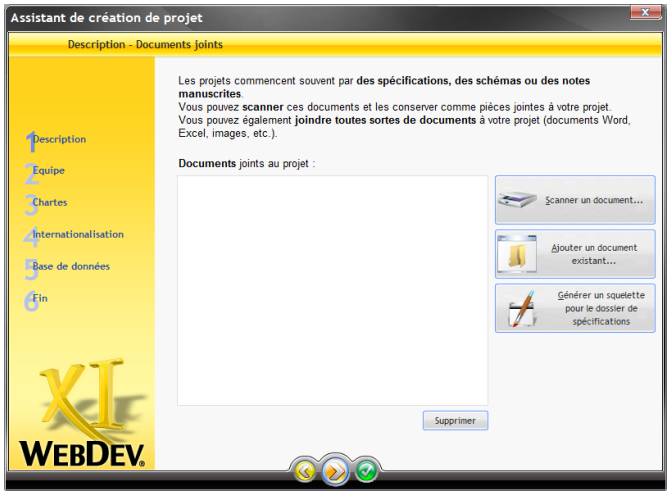
Ce schéma illustre les possibilités offertes au sein de Webdev en ce qui concerne l'interface avec la base de données, c'est celui fourni par PC Soft pour démontrer l'ouverture de leur plateforme composée de Windev-Webdev-Windev Mobile.

2.1.3. Autres informations de projet

Tout au long de la création du projet, un certains nombre d'étapes de création sont suivies.

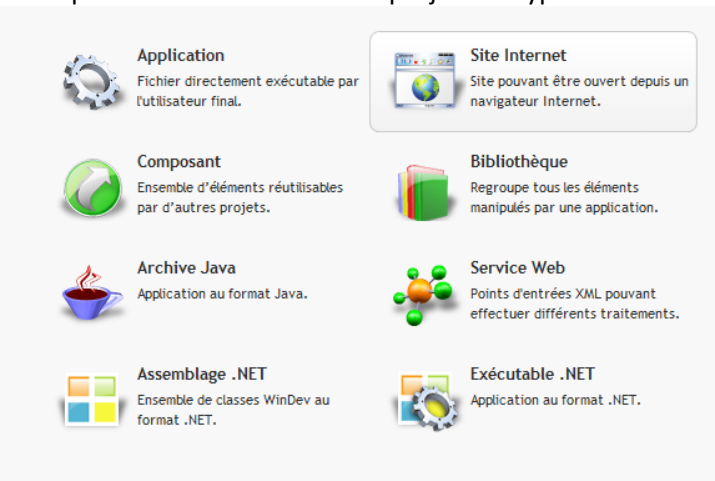
Etape 1 : Description
A travers cette étape de description, on informe le nom souhaité sur le projet, le répertoire de travail (préférablement dans le dossier Mes Sites, créé dans le lecteur C: par défaut).
On peut aussi pour assurer un meilleur suivi de projet et gestion de la documentation, renseigner par une courte phrase le but du projet.

Au moment de la création et à tout moment, on peut choisir d'intégrer des documents tels que des cahiers des charges, des schémas scannés ou tout autre fichier, qui seront utiles dans la réalisation du projet.

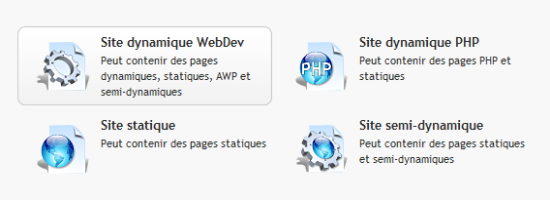


La gestion des documents est donc intégrée au projet.

On doit ensuite informer le type de génération, en clair quel est le type de projet que l'on souhaite obtenir en sortie. Webdev permet de générer des sites Internet ou des composants réutilisables dans d'autres projets. On peut néanmoins ouvrir les projets de types Windev ou Windev Mobile.



Dans notre cas, on choisit Site Internet et on choisit ensuite le type de site qui imposera le type de projet.



Après avoir choisi le type Site dynamique Webdev, on a la possibilité de choisir parmi plusieurs modèles de projet. Ces projets prédéfinis peuvent aider à la création de site, on retrouve par exemple des projets de commerce électronique, de location de DVD ou de gestion de Parc Informatique… Dans notre cas, nous choisissons un projet vierge.

D'autres informations complémentaires peuvent ensuite être complétées concernant le déploiement, le calendrier de réalisation.

Etape 2 : Equipe

C'est à ce moment que l'on peut caractériser la méthode de travail : travailler en équipe ou travailler seul. Dans le cas d'un travail en équipe, il est possible de décrire la liste des équipes qui participeront au projet et qui permettront d'organiser les participants au projet.

Il faut ensuite constituer les équipes choisies en ajoutant les utilisateurs dans les équipes correspondantes.

Pour gérer efficacement les projets, on peut inclure le nouveau projet dans un groupe. Ceci s'avère plutôt utile lors des projets multiples du même ensemble.

Ensuite, on peut choisir d'utiliser ou non le Gestionnaire De Source, vivement conseillé par PC Soft même en utilisation où l'on est seul à développer.

Etape 3 : Charte

On peut choisir la charte de programmation qui détermine alors le préfixage automatique des variables selon leurs types et les noms des champs prédéfinis.

La charte graphique utilisée pour le site peut être également choisie parmi un ensemble de modèles prédéfinis. Ces modèles seront notamment utilisés au niveau des champs.

Etape 4 : Internationalisation
Webdev propose d'intégrer l'internationalisation du site, un projet peut donc supporter plusieurs langues (plus de 20) ce qui est très utile dans l'ensemble du développement.

Etape 5 : Base de données
Cette étape permet de déterminer la base de données qui sera utilisé par le projet. On peut choisir de créer une nouvelle base de données ou d'utiliser une base existante.

2.2. Gestion du projet

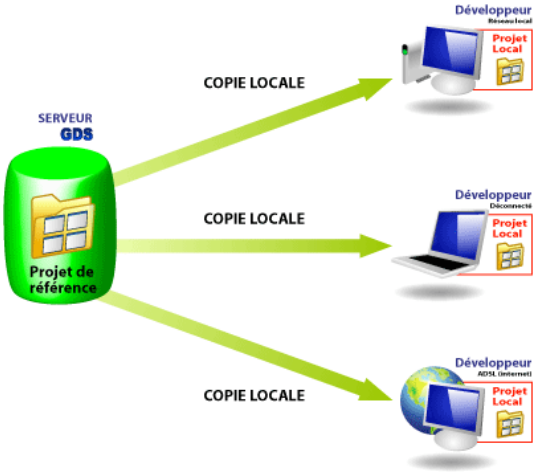
Webdev intègre dans l'ensemble du cycle de développement des outils et fonctionnalités permettant de gérer le projet dans tout le cycle de développement.

L'intégration des documents extérieurs, la gestion du travail en équipe et même les étapes de conception et d'architecture sont intégrées dans le projet au sein de l'éditeur.

2.2.1. Travailler en équipe : le GDS

Le Gestionnaire De Sources (GDS) permet de développer efficacement en équipe, de gérer les historiques et les versions, et d'automatiser la sauvegarde des sources.

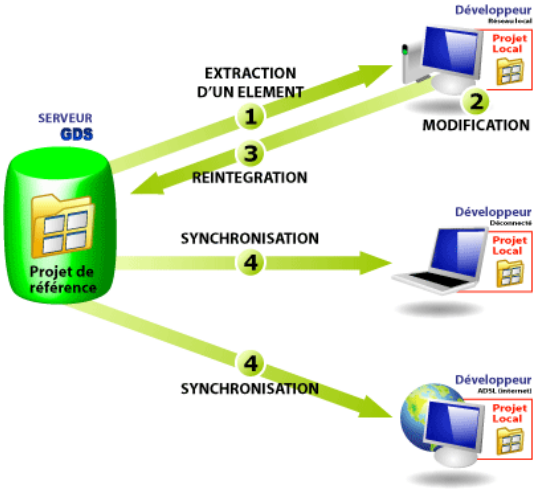
L'aide intégrée de Webdev nous informe de son fonctionnement :



Ainsi on remarque que lors de la première utilisation, le projet de référence situé sur un serveur de gestion de sources est copié sur le poste du développeur.

Il est alors tout à fait possible pour l'administrateur d'établir des droits sur les fichiers et éléments contenus dans le projet pour chacun des développeurs.

Lorsqu'un utilisateur souhaite modifier un élément du projet, on lui propose d'extraire l'élément en question et une fois les modifications effectuées, il peut réintégrer l'élément au sein du GDS.



Bien entendu, ce processus peut intégrer de multiples utilisateurs et la synchronisation peut être bien plus complexe que celle exposée sur ce schéma. Le GDS intègre par ailleurs des processus de vérification d'état avant extraction, de droit de modification et gère intégralement les conflits de versions d'un même élément.
Le GDS intègre également un système de messagerie de projet ainsi que des outils pour attribuer les tâches aux développeurs.

Ajout du projet utilisant le GDS

Afin d'ajouter le projet au gestionnaire de source, on utilise l'assistant associé.
On choisit, si ce n'est pas déjà fait, d'installer une base.

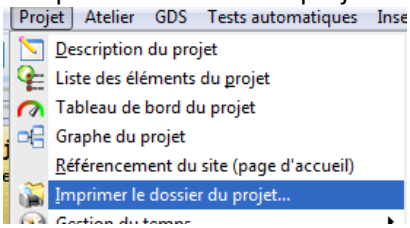
Après avoir renseigné les champs permettant de localiser la base de sources et après que l'on ait cliqué sur « Tester » la connexion (dans le cas d'une base Client/serveur), on peut passer à l'étape suivante d'ajout.

On peut alors voir le répertoire utilisé au sein du GDS pour notre projet, on choisit ensuite les éléments du projet et dépendances à importer (par défaut tous).
On termine l'ajout du projet dans la base de sources.

2.2.2. Documentation

La documentation d'un projet est une étape toujours importante, c'est pourquoi elle est présente en permanence dans Webdev. Ainsi il ne faut pas négliger les descriptions et autres informations sur les éléments que l'on crée puisque ceux-ci sont utilisés pour la génération.

Il est possible d'intégrer les documents tiers dans le projet de façon à fournir des informations complémentaires, et il est également possible de générer la documentation du projet de manière automatique. Pour cela il suffit d'utiliser l'assistant en sélectionnant dans la barre de menu : Projet > Imprimer le dossier du projet…



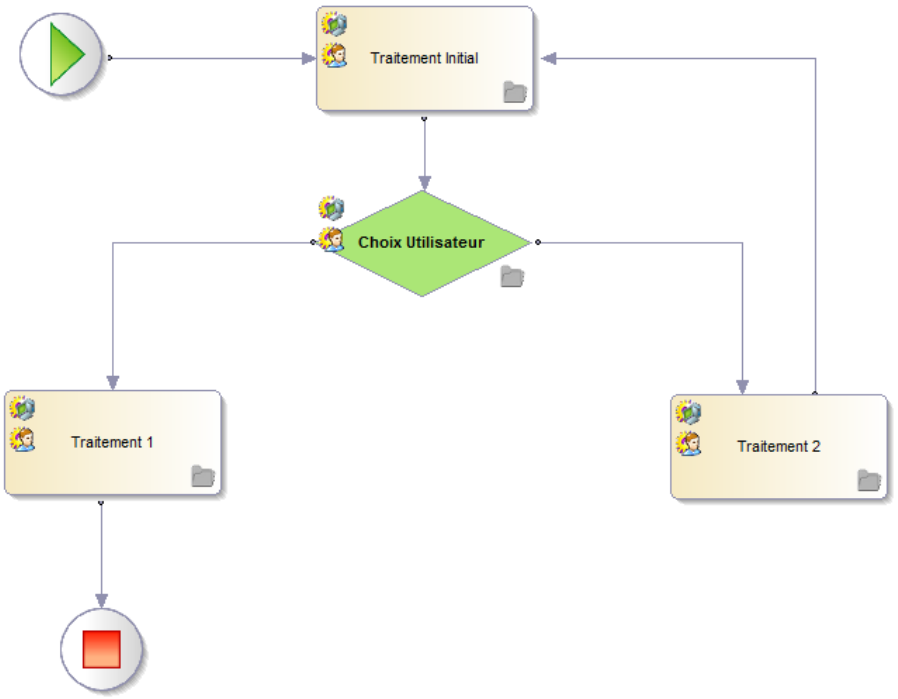
L'assistant guide en différentes étapes invitant à sélectionner le type de dossier, le contenu souhaité et la mise en page et enfin le format de sortie (PDF, HTML, imprimante…).

2.2.3. Architecture et conception

Concernant la conception, Webdev donne la possibilité de compléter le projet à l'aide de diagrammes et de schémas.
Parmi les standards supportés : les diagrammes UML (cas d'utilisation, séquence, état transition, classes…) et des schémas de modélisation souple.

Ces modélisations souples correspondent à un schéma de description de fonctionnement, on y ajoute des étapes, des traitements, des choix.
Cette modélisation peut être liée directement au code et événement dans le projet, les modifications peuvent être répercutées d'un sens comme de l'autre.

C'est un outil puissant de modélisation intégré à Webdev, qui permet de simplifier la compréhension de certains traitements et sont donc fort utiles lors d'un travail en équipe comme lorsqu'on travaille seul pour apporter en maintenabilité.



2.3. Conception de la base de données

Pour ce cas d'exemple, ne pouvant exposer le schéma de la base de données du projet Spiléo, on choisit de créer un schéma d'exemple participant à la compréhension.

2.3.1. Création de l'analyse

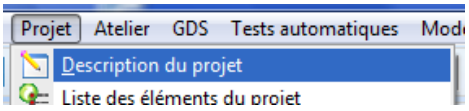
Pour créer l'analyse, l'éditeur de schéma de Modèle Logique de Données (MLD) est utilisé, on peut choisir d'utiliser la génération automatique d'analyse à partir d'un Modèle Conceptuel de Données (MCD) parfois plus simple à comprendre et à réaliser.

Cette création de l'analyse peut s'effectuer à plusieurs moments. Le premier intervient lors de la création du projet, le second se présente lorsqu'on souhaite ajouter une analyse (un schéma de base de données) au projet. Dans les deux cas le même assistant apparaît.

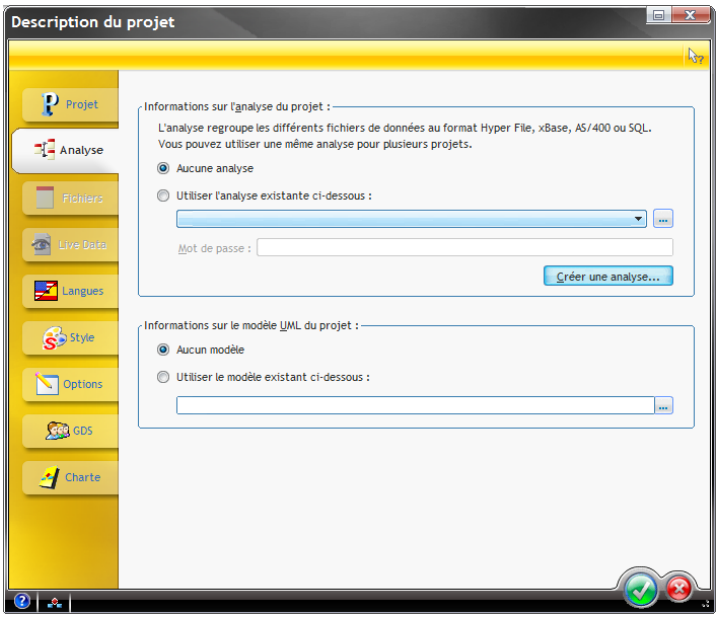
Ajout d'une analyse dans un projet existant

Pour ajouter une analyse au projet, il faut tout d'abord se rendre dans la description (les propriétés) de ce dernier, pour cela on utilise la barre de menu :

Projet > Description du projet :



Après avoir sélectionné l'onglet Analyse dans la fenêtre de description du projet, on choisit de créer une analyse en cliquant sur le bouton associé afin de faire apparaître l'assistant de création et configuration.



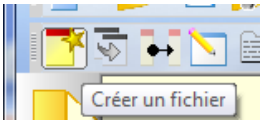
Configuration de l'analyse

La première étape de cet assistant permet de renseigner le répertoire où sera stocké le fichier « .wda » correspondant à l'analyse (le schéma de la base de données). On peut également indiquer une description pour cette analyse afin de renseigner plus en détails cet élément dans le dossier technique du projet.

On peut également imposer un mot de passe pour sécuriser l'accès à cette analyse.

2.3.2. Création des tables

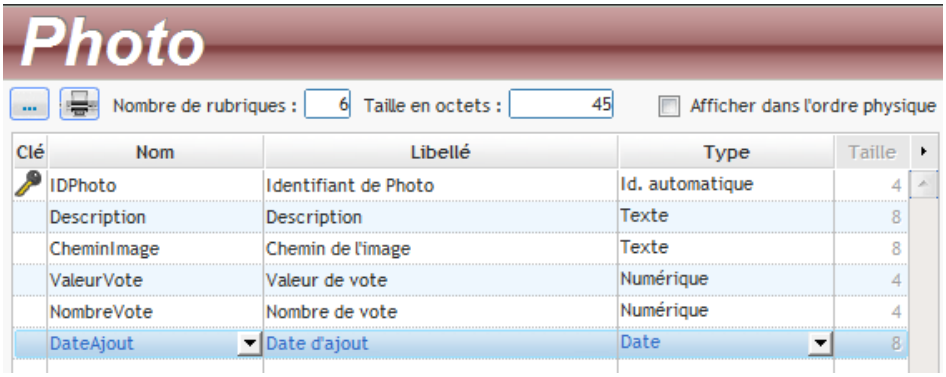
Après avoir validé la création, on se retrouve avec une surface de conception d'analyse vierge.
Pour créer un fichier (une table) de données, on peut procéder de différentes manières :
Clic droit sur la surface> Nouveau fichier
Barre de menu>Insertion>Fichier
Barre d'outils :



L'assistant de création de fichier apparaît, on peut choisir de créer une nouvelle description de fichier, de le créer en s'appuyant sur un modèle existant ou utiliser des fichiers d'une base existante.
On choisit ici de créer une nouvelle description de fichier afin de pouvoir configurer entièrement ce que l'on souhaite dès le début.

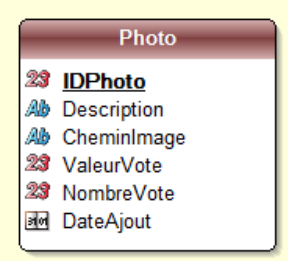
Après avoir renseigné correctement le nom du fichier (Photo) ainsi que le libellé qui sera utilisé dans les assistants de l'éditeur, on prend soin de laisser cochée l'option de création d'identifiant automatique. En effet cet identifiant permet d'effectuer des recherches de manière simple, il sert alors de clé principale.
On choisit ensuite le type de fichier qui va être créé, dans le cas présent : Hyper File Classic et on termine la création.

La fenêtre de création des rubriques du fichier (champs/colonnes de la table) apparaît et on observe déjà la présence d'une première rubrique appelée automatiquement IDPhoto (IDNomDeLaTable de manière générale), on peut lire les informations sur cette rubrique et identifier de suite son type et son utilité.
Pour mettre en avant d'autres fonctionnalités, d'autres rubriques sont ajoutées, pour cela soit on clique dans la table de l'assistant, soit on utilise les icones situés à son bord.
On renseigne le nom et le libellé et on peut ensuite sélectionner son type.



Les types disponibles sont assez explicites dans l'ensemble, ce qui est intéressant c'est de voir que ces types comportent des sous-types permettant d'optimiser les rubriques et donc la base de données, la liste complète et leur utilité sont détaillés dans l'aide.

Une fois toutes les rubriques renseignées et paramétrées, la surface d'analyse se voit ajouter un élément graphique représentant le fichier nouvellement créé (la table).



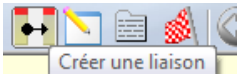
On peut créer directement le schéma MLD complet de la nouvelle base de données via l'éditeur.

2.3.3. Création des liaisons

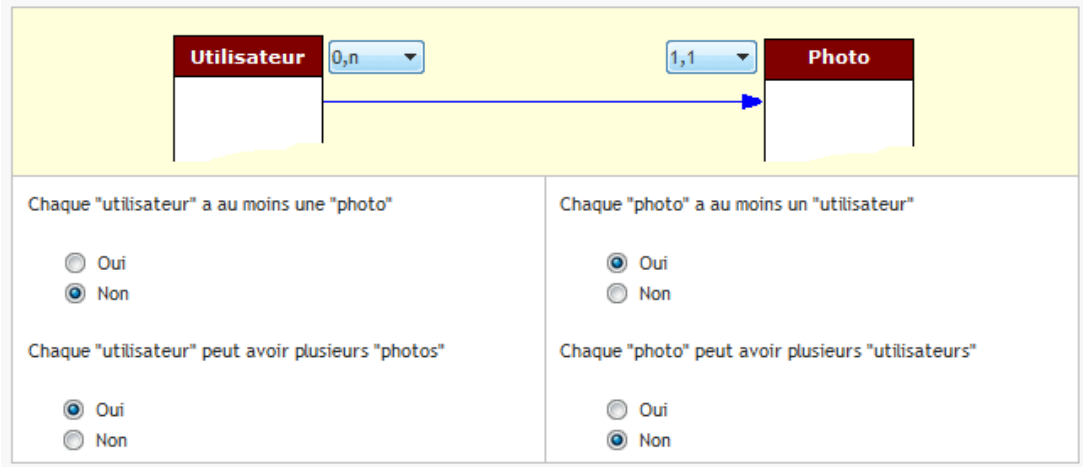
Les fichiers (tables) de la base de données sont souvent liés les uns aux autres.
Pour définir ces liaisons, l'éditeur nous propose un assistant qui aide à définir les liaisons.

Pour définir une liaison :

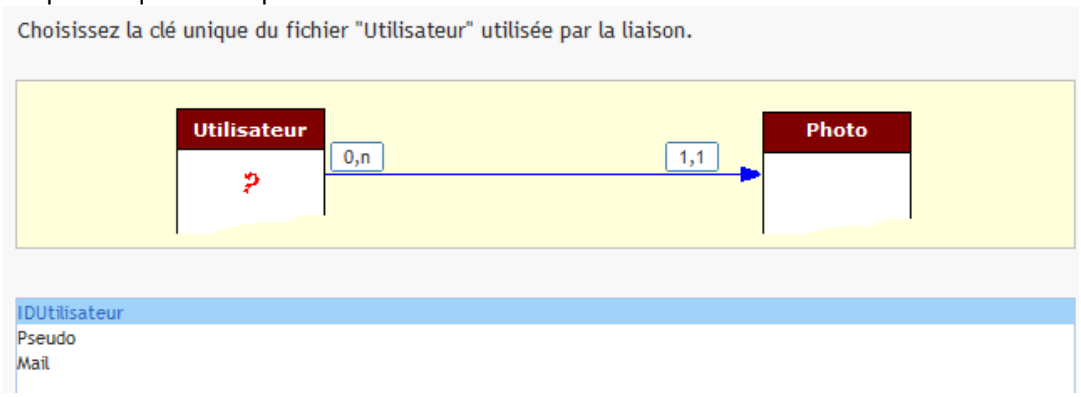
Barre de menu> Insertion> Liaison…
Clic Droit> Nouvelle liaison…
Barre d'outils > Nouvelle liaison



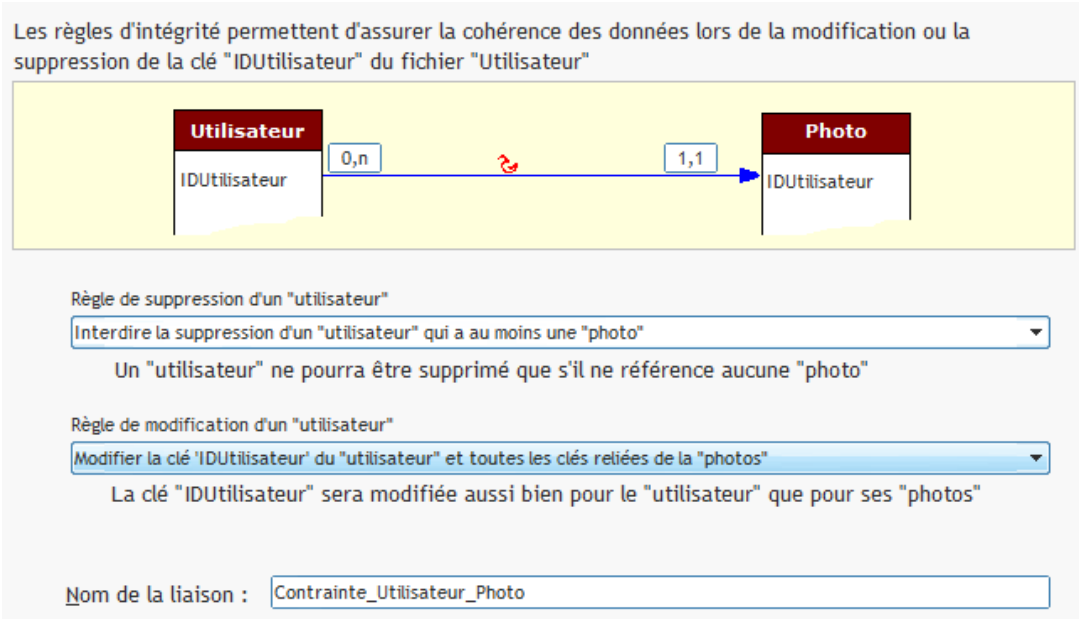
Il suffit ensuite de cliquer sur les fichiers que l'on souhaite lier et l'assistant apparaît et on s'aperçoit que les libellés utilisés pour les fichiers sont utilisés ici, il faut alors définir les cardinalités de la liaison.



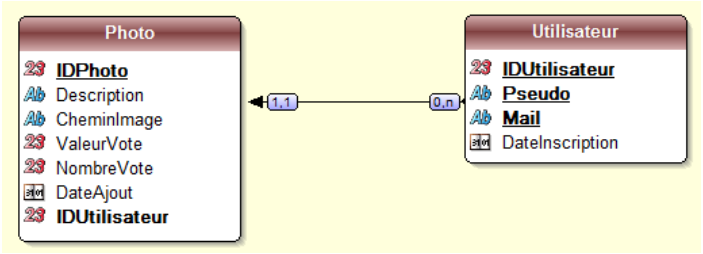
La rubrique clé qui servira pour la liaison



Enfin imposer les règles d'intégrité pour la suppression et la modification de la rubrique de liaison afin de ne pas corrompre la base et assurer la cohérence entre les fichiers, cela créer une contrainte d'intégrité qui sera utilisé par le moteur de sécurité de la base.



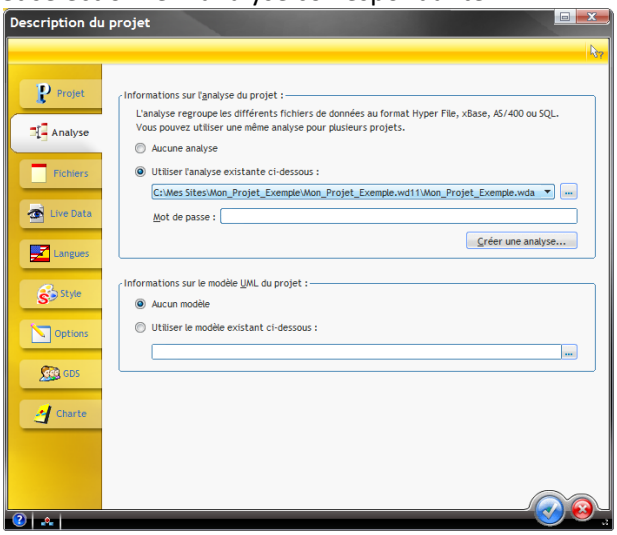
La liaison est ainsi créée et visible directement sur la surface de l'analyse :



Après avoir dessiné le schéma de l'analyse, on est maintenant prêt pour continuer le projet.

2.3.4. Importation d'une analyse existante

Dans le cas d'un développement de plusieurs sites reposant sur la même base de données, il est naturellement indispensable de pouvoir importer l'analyse existant dans les autres projets.
On peut au choix copier l'analyse de l'autre projet, ou directement la référencer depuis son emplacement d'origine.
Pour indiquer l'analyse sur laquelle le projet se base, il faut se rendre dans la description du projet > Onglet Analyse et sélectionner l'analyse correspondante.

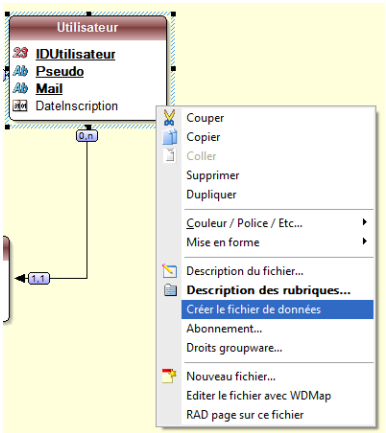


Dans certains cas, il est également possible de directement importer des bases externes. L'exemple de fichier de type MS Access ou Excel, XML,… etc., il suffit ensuite de glisser-déposer le fichier sur la zone d'édition de l'analyse. L'importation est effectuée à travers un assistant.

2.3.5. Création des fichiers physiques

Le schéma de la base de données (l'analyse) est une représentation logique des éléments de la base. Les données sont stockées dans les fichiers physiques (ou les fichiers gérés par le moteur de la base de données).
Pour créer tous les fichiers physiques, on peut utiliser une fonctionnalité de l'éditeur en version 11.

En utilisant le clic droit sur l'objet graphique du fichier sur la surface > Créer le fichier de données.



Une autre solution, dans le code d'initialisation du projet (F2 sur le tableau de bord), utiliser le code suivant qui crée tous les fichiers qui ne le sont pas encore :

```
HCréationSiInexistant("*")
```

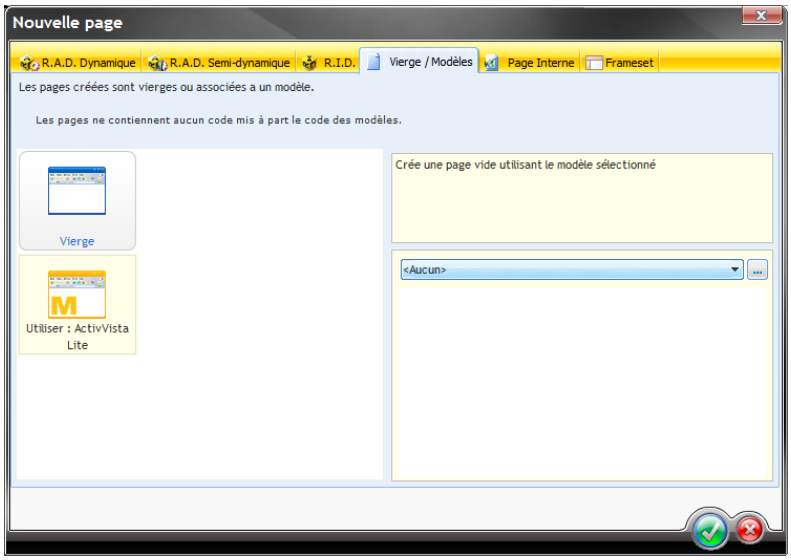
2.4. Création des pages

2.4.1. Création d'une page basique

Pour créer une page dynamique dans le projet :

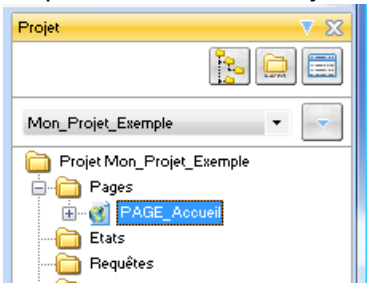
Barre de menu : Fichier > Nouveau
Barre d'outil : Nouveau > Page
Raccourci clavier : CTRL + N

L'assistant de création de page apparaît et on peut sélectionner parmi les différentes options permettant l'automatisation de la création (on sélectionne alors ce qu'on souhaite afficher depuis la base de données, le RAD génère une page avec les champs adéquats) ou on peut choisir de le faire soi même. On choisit également le modèle sur lequel doit être basée la page, cette notion de modèle est abordée un peu plus loin. On choisit « Vierge » pour n'avoir aucun modèle.



L'éditeur affiche une page vierge portant le nom : PAGE_Page1 par défaut (si le nom n'est pas déjà utilisé dans le projet).
On enregistre la page (*CTRL + S ou Fichier > Enregistrer…*) et on lui donne un nom explicite et idéalement préfixé : PAGE_Accueil. Ce nom peut bien évidemment être modifié par la suite et les changements peuvent être diffusés dans le code, en utilisant « refactoring ».

On peut alors observer l'ajout de cet élément du projet dans l'arborescence :



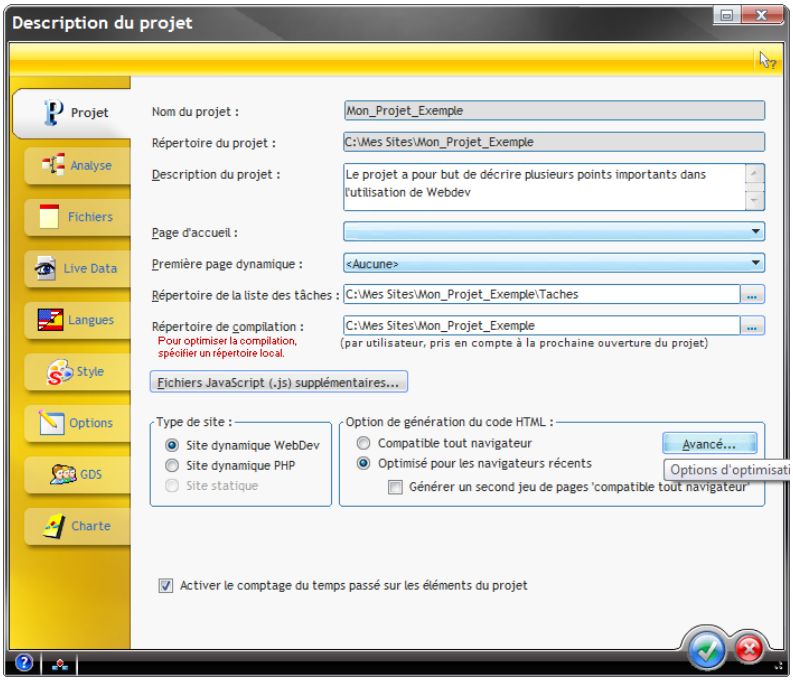
Présentation des composants

Les composants sont des éléments constituant la page comme par exemple des images, des libellés, des zones de saisie. Ces éléments sont traduits en HTML lors de la génération des pages et peuvent donc au besoin être accédés via le DOM à condition de connaître leur propriété Alias (nom de génération HTML).

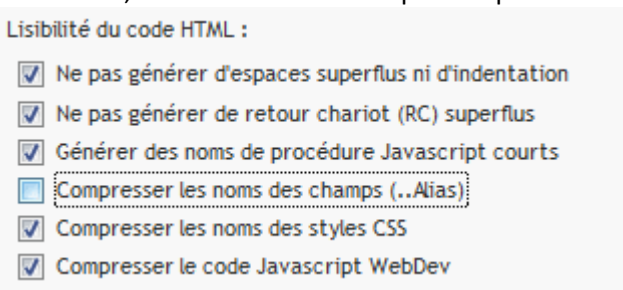
Par défaut ces Alias sont générés automatiquement, et dans l'immédiat, si on souhaite utiliser uniquement du code Webdev, cela ne pose aucun souci puisqu'on peut directement accéder aux champs via leur nom dans l'éditeur.

Si toutefois on souhaite avoir des noms compréhensibles dans le code de sortie, il faut le spécifier dans les options de génération dans la description du projet.

Barre de menu > Projet > Description du projet.



Par défaut, le code de sortie est optimisé pour avoir un code HTML le plus concis possible.

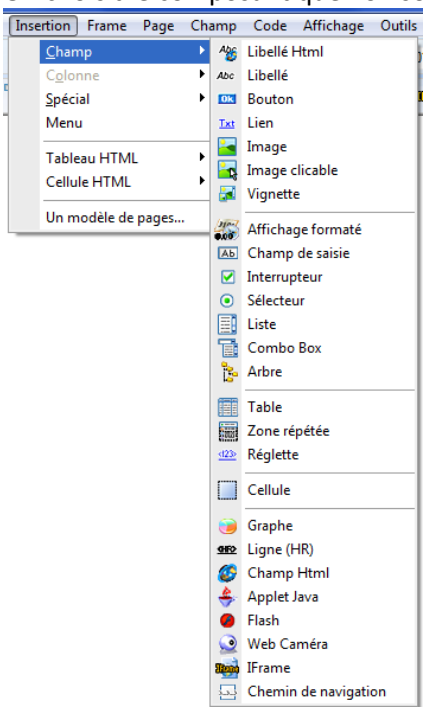


Ajout de composants

Pour ajouter un composant sur une page, on peut utiliser :

Barre de menu > Insertion

On choisit le composant que l'on souhaite parmi la liste :



La barre d'outils associée : un simple glisser-déposer pour ajouter un élément à la page :

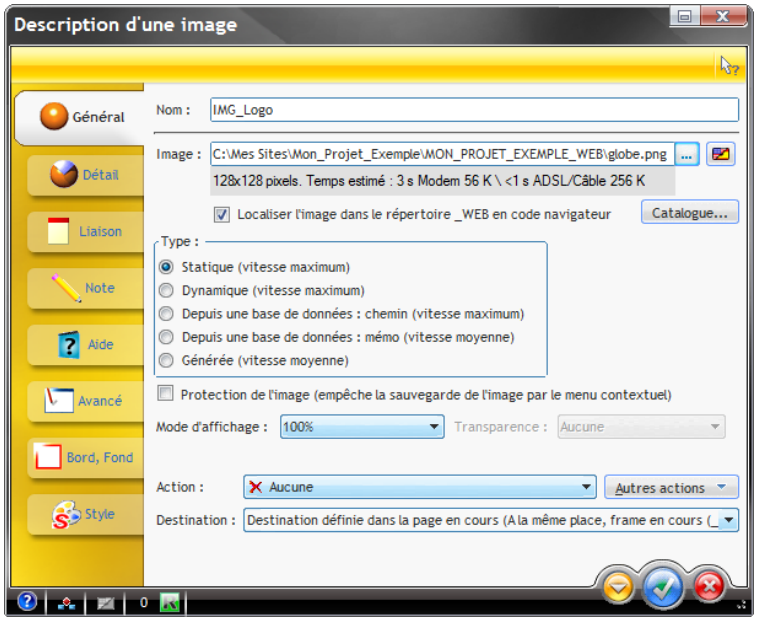
Des fonctionnalités peuvent également aider à la disposition des éléments sur la page, notamment les règles et les aides au positionnement/alignement.

On peut s'aider en utilisant les touches F6 et F7 pour disposer les champs sur la page.

Modification des propriétés des composants

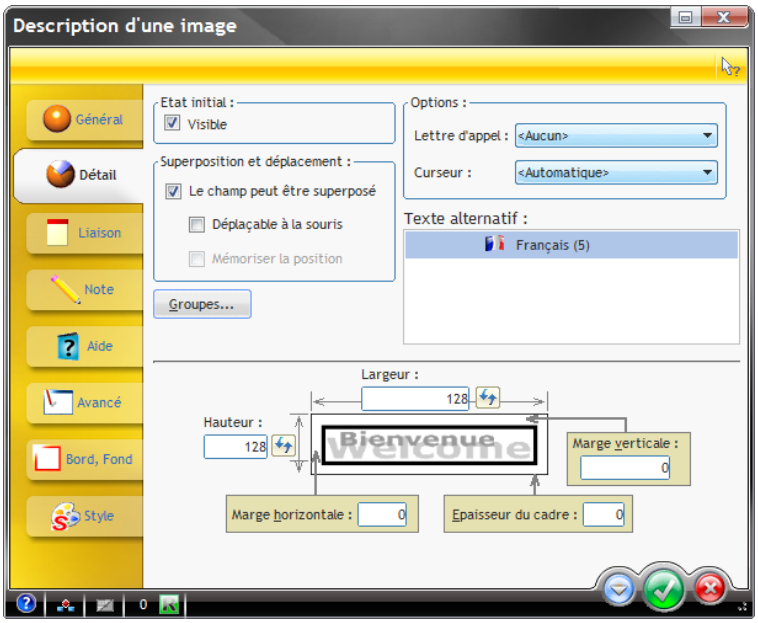
Après avoir ajouté les composants, il est primordial de définir leurs propriétés, pour cela Webdev utilise la technique des onglets réunis dans un assistant de description du champ.
Ici on désire modifier un champ image inséré sur la page :

Onglet Général :



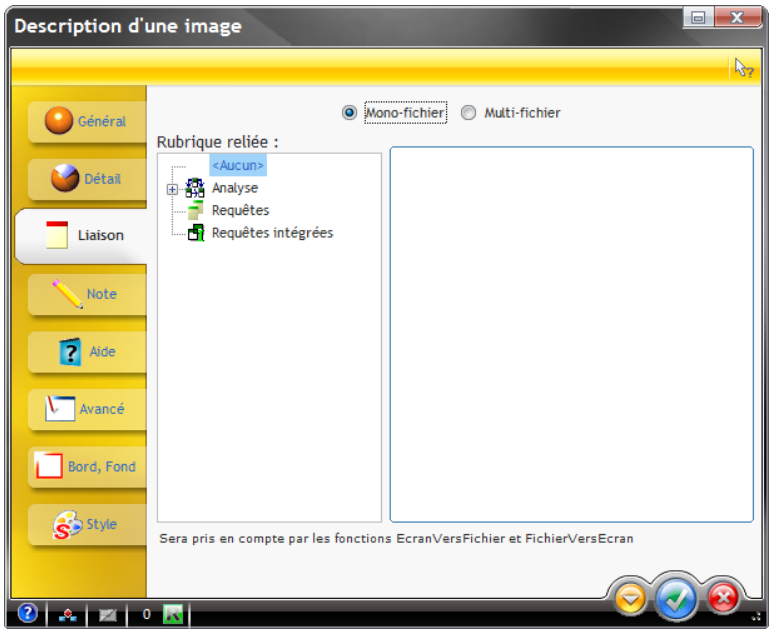
On y modifie le nom, le chemin de l'image qui doit, pour être exploitable en production, être un chemin accessible par et via le serveur web.
On peut voir d'autres propriétés comme le type, le mode d'affichage et les actions associés au clic sur ce premier onglet Général.

Onglet Détail :



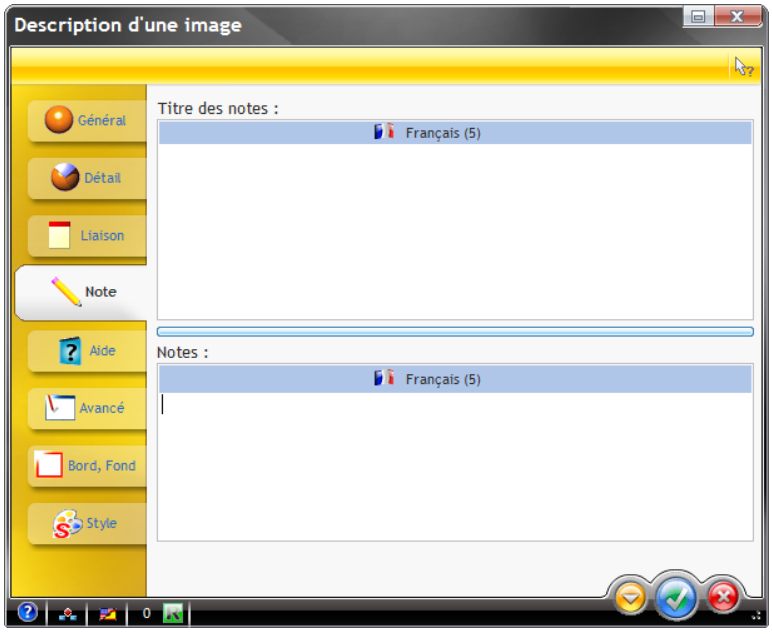
On peut indiquer l'état de visibilité, la possibilité de superposition (le champ est alors contenu dans une cellule HTML (DIV)). De même pour les dimensions et les options d'appel au clavier et de curseur. Il est important de remarquer que c'est également dans cet onglet que figure le texte alternatif de l'image, utile pour le référencement.

Onglet Liaison :



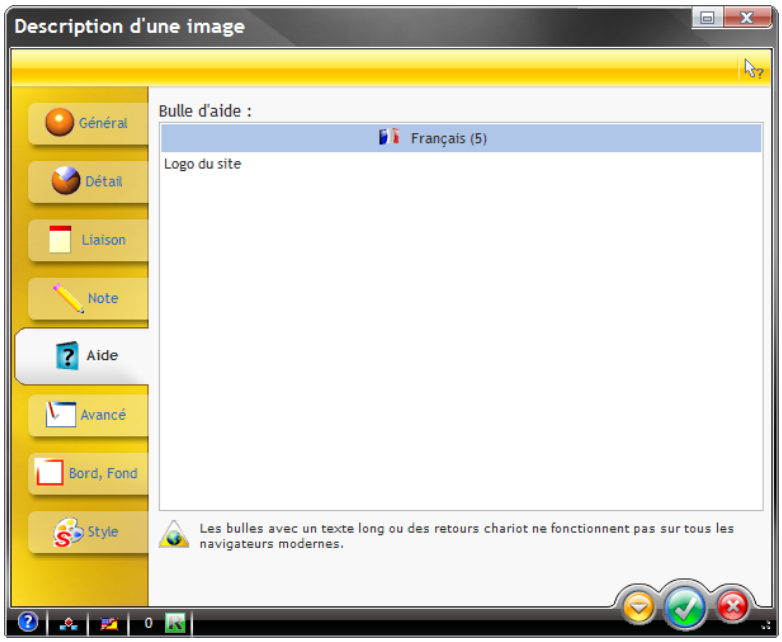
C'est à travers cet onglet que l'on gère le binding automatique des champs à la couche de données. Cet aspect sera abordé dans la partie d'interaction avec la base de données dans la suite de cet article.

Onglet note :



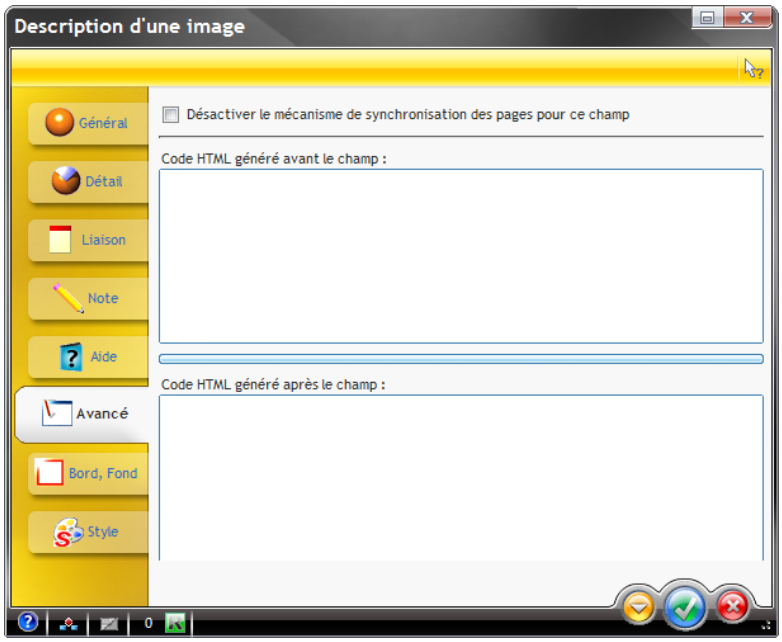
Les notes sont utilisées dans le dossier technique et peuvent être utiles pour le développeur.

Onglet Aide :



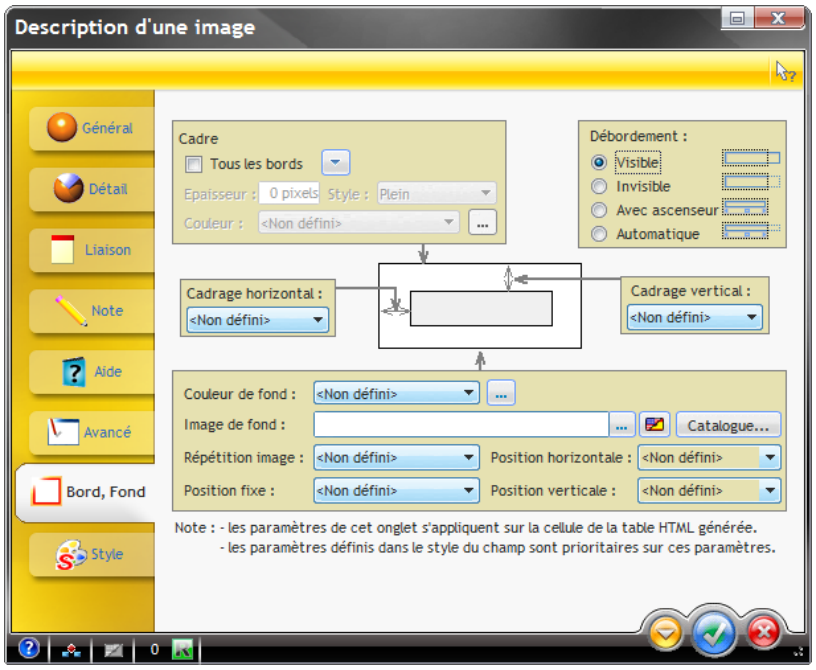
Il est possible d'indiquer l'information qui sera affichée dans la bulle d'aide qui se présente au survol de la souris par-dessus le champ.

Onglet Avancé :



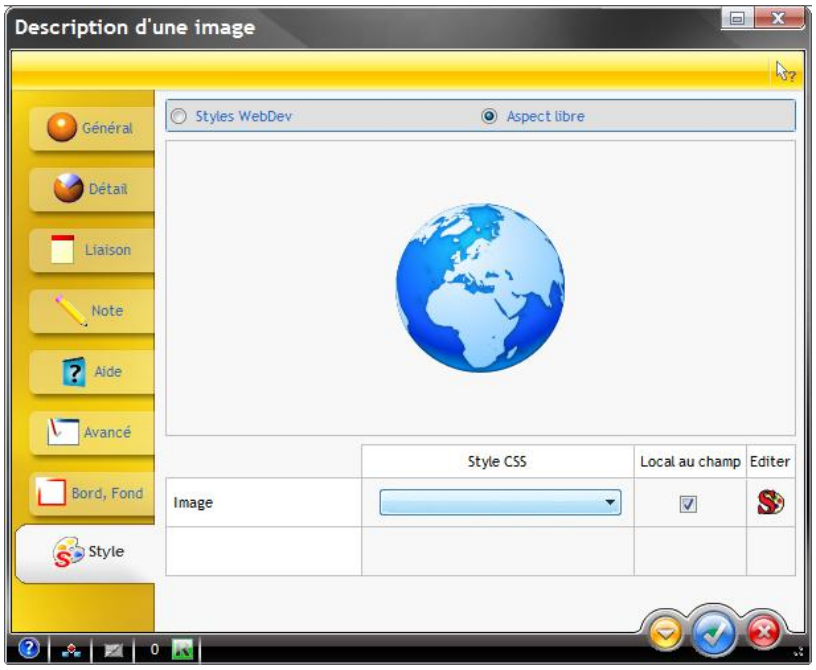
Webdev permet d'accéder quelque peu au code de sortie afin de pouvoir y ajouter ce que l'on souhaite, ainsi au besoin on peut ajouter du code HTML avant ou après le champ.

Onglet Bord, Fond :

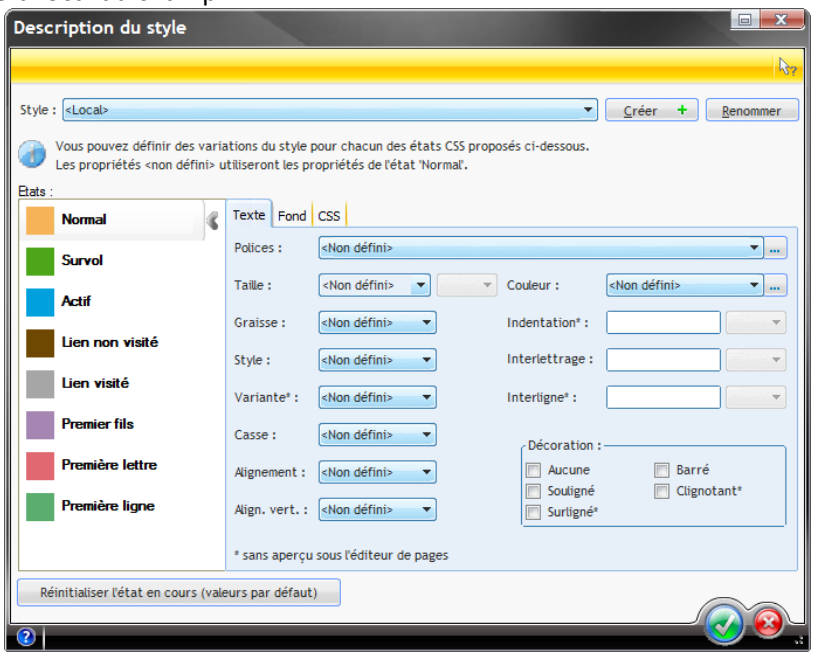


On y édite une partie des propriétés CSS, notamment le cadre (border en css), le débordement (overflow), le cadrage (padding) ainsi que le fond (background).

Onglet Style :



Les styles CSS sont gérés par défaut par Webdev, on peut d'ailleurs les gérer en se basant sur des styles prédéfinis (styles Webdev) ou choisir d'utiliser un aspect libre et s'appuyer alors sur un style qui sera local au champ.

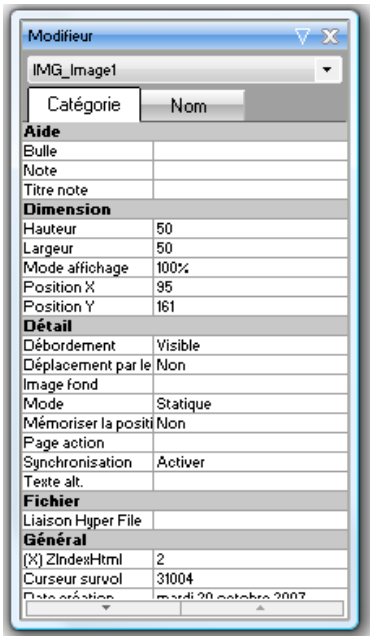


Il s'agit véritablement du style CSS appliqué, on peut définir toutes les propriétés CSS (pas encore établies dans d'autres onglets) comme par exemple les formats de texte (graisse, style, police…) en fonction des états (survol, actif…).

Le modifieur :

On peut également modifier toutes ces propriétés grâce à un modifieur affichable via :

Barre de menu : Affichage > Barre d'outils > Modifieur



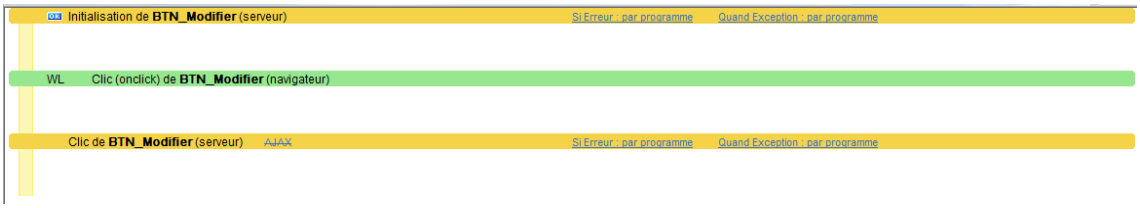
Ici le champ dont on change les propriétés était une image, la plupart des propriétés sont communes aux champs, certaines peuvent être spécifiques comme par exemple l'état (Actif, Grisé, Affichage seul) et bien d'autres. Toutes réunies dans la fenêtre de description regroupant les propriétés par onglet.

Exploitation de ces composants

Une fois les champs ajoutés et correctement configurés, il est temps de s'en servir dans le code des traitements.

Pour accéder au code correspondant à un traitement sur un champ :

Clic droit sur le champ > Code
Raccourci clavier : F2



Ici sur le bouton BTN_Modifier, le passage dans l'éditeur de code nous amène dans l'événement de clic côté Serveur.

On peut remarquer la présence d'un autre code serveur correspondant à l'initialisation du champ et enfin l'événement de clic du coté navigateur.
Cette notion de code serveur et code navigateur sera plus complètement décrite par la suite mais, pour résumer, le code navigateur est celui qui s'exécute sur le poste client et qui est donc un traitement JavaScript.

Le traitement souhaité sur le clic de ce bouton est simple, on saisit un texte dans un champ de saisie (SAI_Modif) et on souhaite que lorsqu'on clique, on affiche le texte tapé dans un champ libellé prévu à cet effet (LIB_TexteAModifier).
Voici donc le code associé au traitement de clic du côté serveur :

```
SI SAI_Modif <> "" ALORS
	LIB_TexteAModifier..Valeur = SAI_Modif..Valeur
SINON
	Info("Le champ de saisie est vide")
FIN
```

La notation peut toujours surprendre, elle n'en demeure pas moins très compréhensible.
Dans un premier temps on vérifie que le champ de saisie n'est pas vide.
S'il n'est pas vide, on modifie la valeur du libellé par la valeur contenue dans le champ de saisie.
S'il est vide on affiche une information au visiteur (pas très esthétique pour le moment).

Les propriétés d'un champ ou d'un élément quelconque du projet peuvent être accédés en code via la forme suivante : (attention, certaines d'entre elles ne sont pas accessibles en écriture)

```
NomDuChamp..NomPropriété
```

On dispose d'une aide à la saisie efficace pour ces propriétés.

2.4.2. Création d'un modèle

La notion de modèle est simple à saisir, c'est une base qu'utilise une page pour ses styles CSS mais aussi pour les champs communs à plusieurs pages.
Equivalents des Masters Pages dans d'autres plateformes, les modèles de pages WebDev permettent de gagner en cohérence entre les pages, en temps de développement et en maintenabilité.

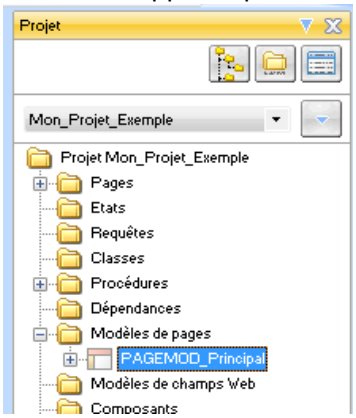
Très utile pour les menus et parties communes des pages, on n'édite les champs et propriétés qu'à un seul endroit pour impacter toutes les pages utilisant le modèle.

Pour créer un modèle :

Barre de menu > Nouveau > Modèle
Barre d'outils > Nouveau > Modèle
CTRL + N > Modèle

Le modèle apparaît alors dans l'éditeur (on remarque le bord jaune de l'éditeur indiquant qu'il s'agit d'un modèle), on peut l'enregistrer de suite en lui donnant un nom, toujours préfixé pour profiter de l'aide à la saisie dans l'éditeur de code, mais aussi pour mieux s'y retrouver en tant que développeur et favoriser ainsi le travail en équipe.

Ce modèle apparaît par défaut dans l'arborescence du projet dans le sous dossier Modèles :



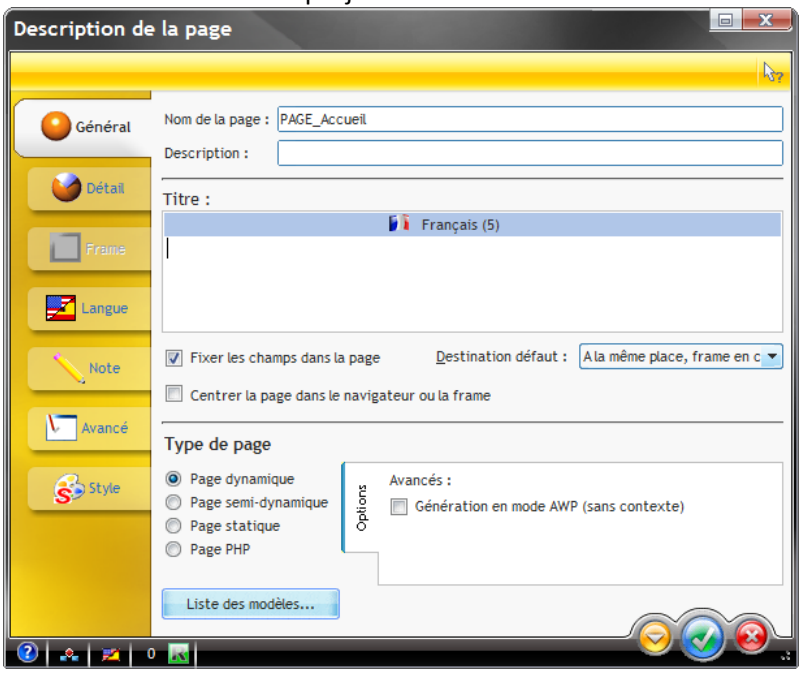
Pour ajouter des champs au modèle, on procède comme sur une page classique, de même pour les propriétés.

Pour la disposition, on dispose des mêmes aides pour placer les éléments. On peut également mettre en œuvre ce qui est appelé dans Webdev « les zones extensibles ». Cela permet par exemple de disposer des éléments toujours en bas du navigateur (les positions sont établies en pourcentages).
Pour afficher ces zones extensibles : *Affichage > Zones extensibles (CTRL + I)*.
De même pour afficher les différentes tailles de navigateur : *CTRL + M.*

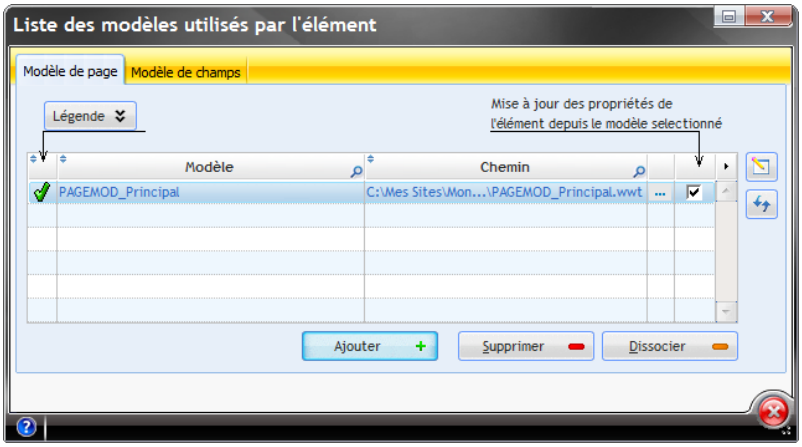
Dans certains cas, on a créé une première page qui nous plait et dont on aimerait se servir pour établir un modèle. Webdev permet bien sûr de réaliser cette tâche simplement :
Barre de menu > Fichier > Enregistrer comme un modèle…

2.4.3. Utilisation du modèle au sein des pages

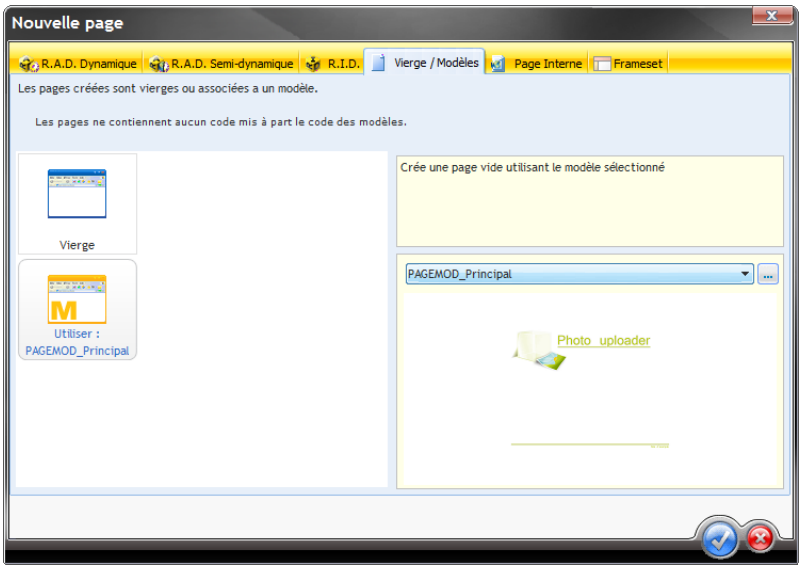
Pour utiliser un modèle dans une page existante, il faut se rendre dans la fenêtre de description de la page et afficher la liste des modèles du projet :



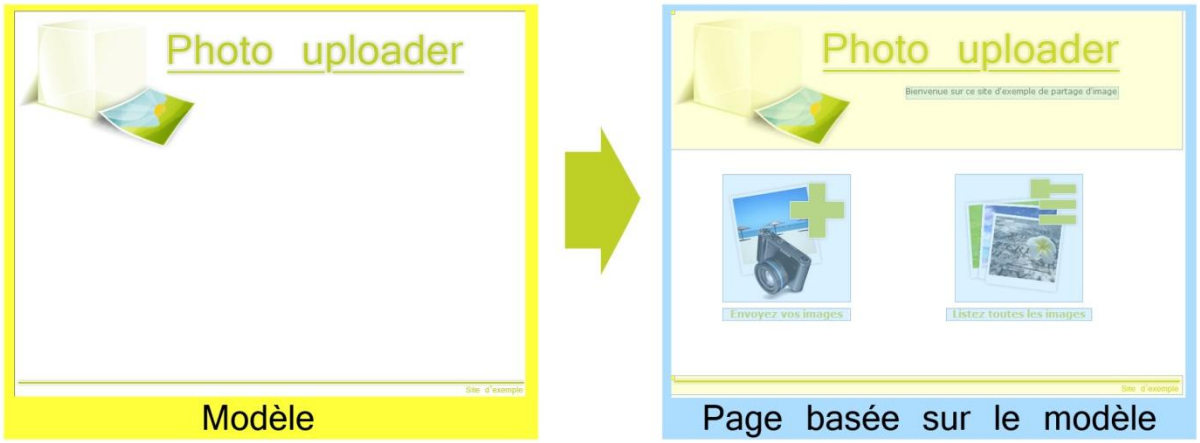
La liste s'affiche, et on peut ajouter les modèles ou en retirer selon différentes options. Les conflits de positionnement des champs qui peuvent survenir sont signalés dans l'éditeur par un message et par une indication visuelle dans la surface d'édition.



Lorsqu'on créé une nouvelle page, on peut choisir de baser cette dernière directement sur le modèle que l'on souhaite :



Explication avec la page d'accueil du site : PAGE_Accueil



Le modèle impose deux champs à la page basée sur le modèle, affichés en jaune sur la partie droite du schéma. Les champs bleus sont ceux propres à la page.

Sur l'éditeur, les champs issus du modèle sont indiqués dans l'éditeur par un petit carré jaune dans le coin haut gauche, pour ouvrir le modèle correspondant rapidement, il est possible de faire un clic droit et de choisir d'ouvrir le modèle.

2.5. Développement avancé de pages

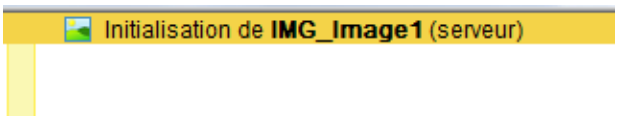
2.5.1. Code serveur, navigateur et AJAX

Abordé plus haut, on observe différentes parties dans le code : serveur et navigateur.

Code serveur

Le code serveur est celui qui s'exécute sur le serveur dans le moteur de page. Ce code est très peu limité concernant l'interaction avec les données et les traitements qui peuvent être effectués sur le serveur.

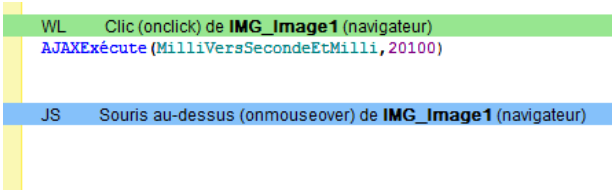
Dans l'éditeur le code serveur est indiqué en orange et par une précision entre parenthèses.



Code navigateur

Le code navigateur est le code qui s'exécute sur le poste client, il s'agit d'un code W-Langage ou JavaScript. Le code W-Langage navigateur est d'ailleurs traduit en JavaScript dans le code de sortie.

Dans l'éditeur le code navigateur est indiqué en vert (W-Langage) et bleu (JS) et par une précision entre parenthèses. Le type de code peut être modifié en cliquant sur l'abréviation du bandeau.



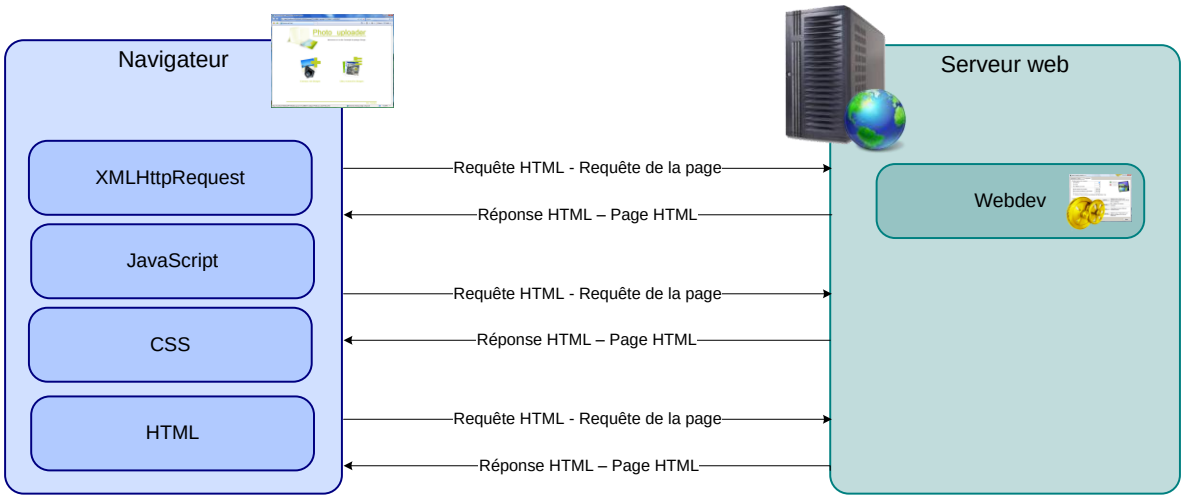
La génération de ce code peut être paramétrée dans les options de génération dans la description du projet. A travers ce code on peut accéder à des informations relatives au navigateur et effectuer des traitements d'affichage.

Utilisation d'AJAX

Webdev permet également d'utiliser la technologie AJAX pour effectuer des traitements serveur sans nécessiter d'effectuer un postback complet de la page HTML pour afficher le résultat sur la page ou tout simplement communiquer avec le serveur.
Basiquement, on utilise un objet (XMLHttpRequest) pour transporter des requêtes et les réponses associées en XML en utilisant JavaScript.

Le schéma suivant illustre les échanges entre le navigateur et le serveur web lors d'appels classiques n'utilisant pas la technologie AJAX.

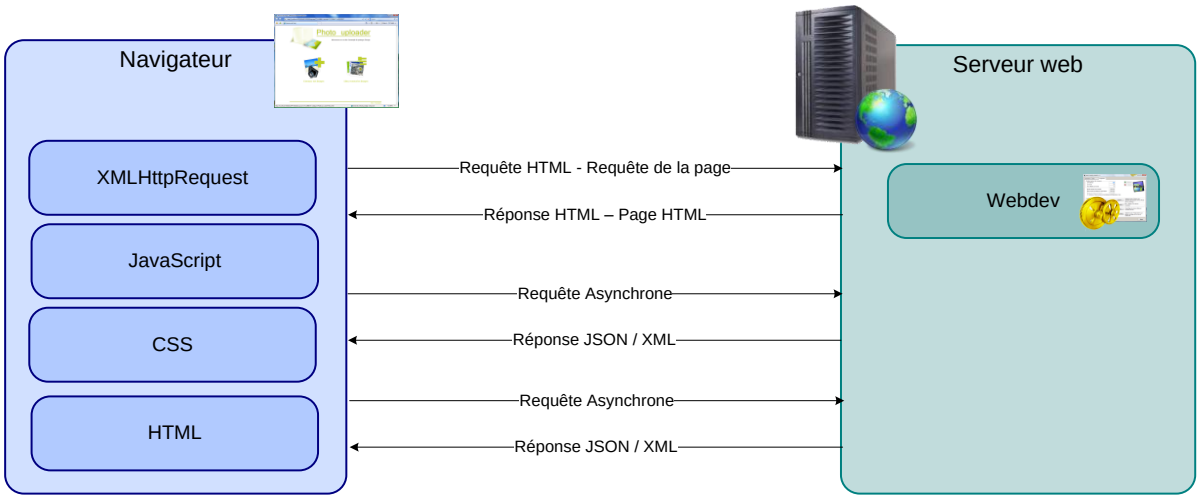
APPELS CLASSIQUES – NON AJAX



A chaque requête effectuée correspond un rendu complet de la page et le navigateur recharge l'intégralité des champs, ce qui peut être gênant pour l'utilisateur.

Dans le cas d'appels AJAX, voici le schéma qui correspond :

APPELS AJAX



La première requête reçoit une réponse HTML (la page est alors complètement chargée).
Lors de la seconde requête qui peut intervenir depuis un événement du navigateur ou depuis le serveur, la réponse contient des données en JSON (JavaScript Object Notation) ou XML (Extensible Markup Language) qui sont alors traitées en JavaScript (via l'objet XMLHttpRequest) et qui ne nécessitent pas un rafraichissement complet de la page mais mettent à jour uniquement la zone en question quand il le faut.

L'expérience utilisateur n'en est que plus agréable et les transferts entre le navigateur et le serveur web sont réduits dans la plupart des cas. Volontairement, je ne descends pas plus en détails sur la technologie AJAX mais il est important de saisir parfaitement les avantages et inconvénients que son utilisation peut apporter pour être conscient des limitations et des effets.

Pour utiliser AJAX au sein de Webdev, PC Soft a véritablement simplifié la chose.

Sur un traitement serveur associé à l'événement d'un champ :
Sur un champ dans un code habituellement serveur, pour activer AJAX il suffit de cliquer sur l'indication dans l'éditeur de code, exemple ici avec un code de clic serveur sur un bouton :

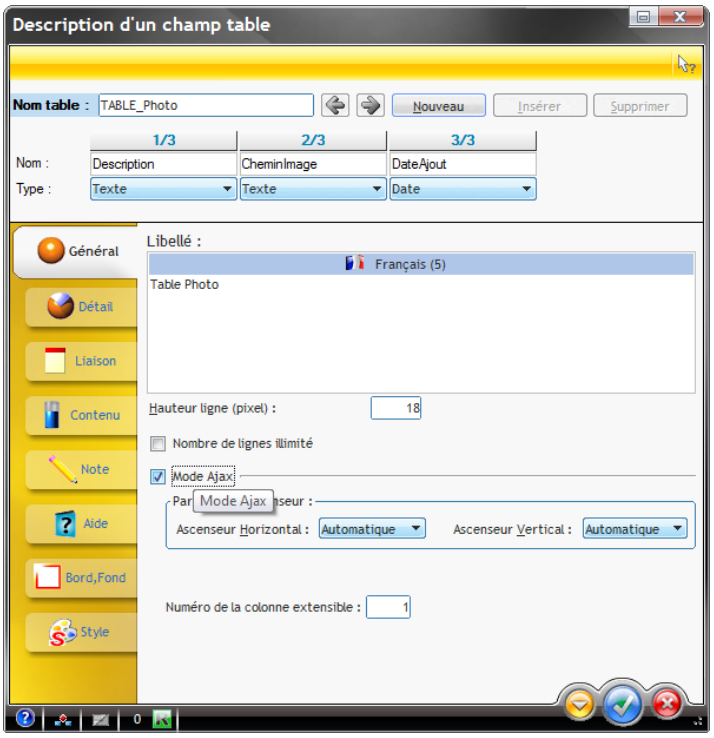
```
Clic de BTN_Modifier (serveur)    AJAX Activé
SI SAI_Modif <> "" ALORS
    LIB_TexteAModifier..Valeur = SAI_Modif..Valeur
SINON
    Info("Le champ de saisie est vide")
FIN
```

Le traitement AJAX est alors activer sur le clic sur le bouton.
Pour ce traitement simple on peut estimer que c'est un inutile, mais lorsqu'on doit interroger une base de données à partir des données entrées sur la page et renvoyer une réponse sans rafraîchir complètement la page, cela s'avère fort pratique.

Sur un champ directement :
Certains champs peuvent être enrichis par l'expérience AJAX, c'est le cas notamment des tables, des zones répétées ou des combos.
Cela permet par exemple dans le cas des zones répétées, de ne charger que les éléments visibles sur la page, au défilement dans la zone répétée, les données sont chargées dynamiquement.

Dans les tables qui utilisent AJAX, lors de l'édition dans ces dernières, les données sont chargées dynamiquement et modifiées dans la base de manière transparente (au souhait du développeur).

Pour activer AJAX sur un champ, il faut se rendre dans sa fenêtre de description et cocher Mode Ajax, ici activé sur un champ table :



Le mode AJAX est disponible sur la plupart des contrôles disponibles, la liste ne cesse de s'étoffer au fil des versions de Webdev.

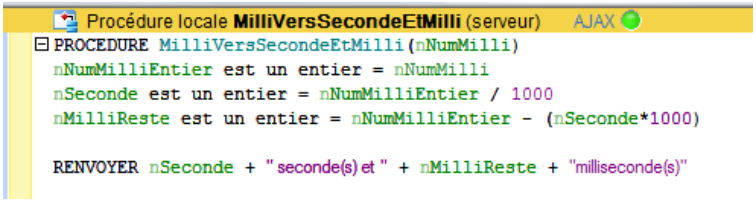
2.5.2. Procédures

Lorsque l'on effectue le même traitement à plusieurs endroits dans une page, on peut utiliser les procédures locales pour regrouper les instructions.

Lorsqu'un même traitement est effectué dans plusieurs pages, on utilise les procédures globales qui sont établies au niveau du projet. Ainsi depuis n'importe quel élément du projet, on peut utiliser les procédures globales créées.
Ces procédures globales peuvent être regroupées en collection permettant une meilleure architecture et une réutilisabilité améliorée pour d'autres projets par exemple.

Ces procédures globales comme les procédures locales comportent naturellement deux types : serveur et navigateur.

Les procédures serveur (globales ou locales) peuvent être appelées en AJAX, pour cela il faut que l'appel AJAX soit autorisé pour ces procédures, pour l'activer il suffit d'activer AJAX sur la fonction :



Pour appeler cette fonction depuis un code W-Langage navigateur :

```
AJAXExécute(MilliVersSecondeEtMilli,20100)
```

De manière générale :

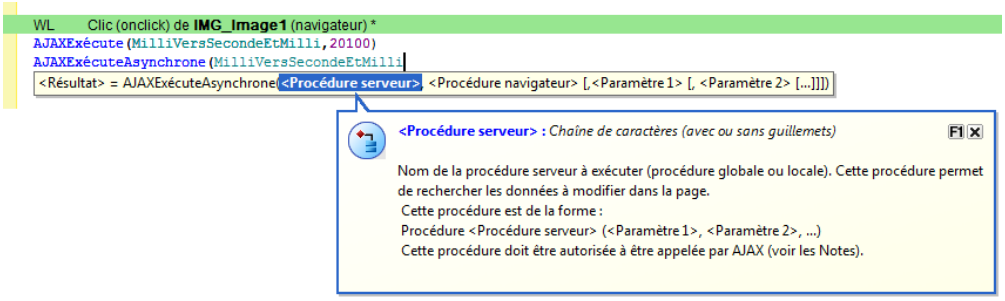
```
AjaxExécute(NomProcédure, Paramètre1, …)
```

Ou

```
AjaxExécuteAsynchrone(NomProcédureServeur, NomProcédureNavigateur, Paramètre1,…)
```

La différence entre ces deux méthodes d'appel est que dans le premier cas, le traitement n'est pas effectué en tâche de fond, le code qui suit éventuellement le *AjaxExécute*() n'est exécuté qu'après réponse du serveur.
Dans le second cas, le traitement est effectué en tâche de fond, la réponse n'est pas attendue et le reste du code s'exécute sans attente. Il faut alors indiquer la procédure navigateur qui sera appelée une fois la réponse du serveur reçue (fonction de callback).

Dans les deux cas, comme en permanence dans l'éditeur, l'aide à la saisie nous indique le prototype ainsi qu'une information complémentaire sur les paramètres des fonctions.



2.6. Interaction avec la base de données

Dans le cas des sites dynamiques, il est possible d'interagir avec une base de données.
Pour cela il faut utiliser un traitement serveur qui peut alors utiliser différentes techniques pour accéder à la base de données : les fonctions HyperFile du W-Langage, l'utilisation des requêtes de Webdev, et les requêtes SQL.

2.6.1. Utilisation des fonctions HyperFile

Le W-langage simplifie l'accès aux données, bénéficiant d'une aide à la saisie efficace et doté de nombreuses fonctions utilisables. Les fonctions HyperFile sont préfixés par «H » comme ceci : HNomFonction().

Recherche dans un fichier de données

Un exemple de recherche dans un fichier dans le code de clic serveur d'un bouton :

```
HLitRecherche(Utilisateur,Pseudo,SAI_Pseudo)
SI HTrouve(Utilisateur) ALORS
	SI Utilisateur.MotDePasse = SAI_Mot_de_passe ALORS
		//code de traitement...
		//
		Info("Vous êtes bien authentifié")
	FIN
FIN
```

- Dans un premier temps, on effectue la recherche sur le fichier Utilisateur de l'analyse (table de la base de données) :
  La fonction HLitRecherche() permet de positionner l'index de lecture sur l'enregistrement trouvé en effectuant la recherche (*HRecherche*()) et d'initialiser le contexte de données (*HLit*()). Elle prend en paramètres : le nom du fichier de recherche, la rubrique sur laquelle la recherche est effectuée, la valeur recherchée. Elle prend un dernier paramètre correspondant au type de recherche effectué et aux options de la recherche.

- On teste si la recherche aboutit à un résultat à l'aide de la fonction *HTrouve*() qui renvoie un booléen indiquant cette information.

- On teste ensuite si le mot de passe correspond à celui de l'enregistrement en cours dans la base de données. Pour accéder à la valeur de la base de données, puisque l'index de lecture est initialisé, on peut utiliser la notation suivante : *NomDuFichier.NomRubrique*.
  La valeur renvoyée est véritablement celle lue dans la base de données au moment de la recherche.

Recherche avancée dans un fichier de données

Le code suivant permet de récupérer des éléments dans un fichier dont une rubrique commence par la chaîne indiquée, c'est un code serveur :

```
sMaClé est une chaîne = HFiltreCommencePar(Utilisateur,Pseudo,"to")

HLitPremier(Utilisateur,sMaClé)
TANTQUE PAS HEnDehors()
       Info(Utilisateur.Pseudo)
       HLitSuivant()
FIN
//Annule l'effet du filtre
HDésactiveFiltre(Utilisateur)
```

- On établit le filtre en utilisant la fonction *HFiltreCommencePar*().
  Cette fonction prend en paramètre le fichier sur lequel le filtre doit être appliqué, la rubrique et la chaîne permettant d'effectuer la condition.
  Elle renvoie la rubrique clé sur laquelle le filtre est appliqué.

- On positionne l'index de lecture sur le premier enregistrement du fichier en indiquant que la rubrique de parcours est celle que l'on utilise dans le filtre.

- Dans une boucle qui se répète tant qu'on atteint pas la fin de fichier, pour vérifier si on n'est justement pas en dehors du fichier, on utilise *HEnDehors*() qui renvoie un booléen correspondant à l'information.
  On affiche les données lues en base.
  On positionne ensuite l'index de lecture sur le prochain enregistrement du fichier en utilisant *HLitSuivant*().

- On désactive le filtre appliqué en fin de traitement, pour éviter qu'il ne s'applique sur d'autres éléments de code (ici accessoire).
  Pour cela on utilise la fonction *HDésactiveFiltre*() qui prend en paramètre le fichier sur lequel le filtre est établi.

Ajout dans un fichier de données

Un exemple d'ajout simple dans un fichier, toujours dans le code de clic serveur d'un bouton :

```
Utilisateur.Pseudo = SAI_Pseudo
Utilisateur.MotDePasse = SAI_Mot_de_passe
Utilisateur.Mail = "toto@mail.com"
Utilisateur.DateInscription = DateSys() + HeureSys()
SI PAS HAjoute(Utilisateur) ALORS
	Info(HErreur(hErrEnCours))
FIN
```

- On initialise le contexte de données « manuellement » :
  NomDuFichier.NomRubrique = valeur à associer
  On utilise *DateSys*() et *HeureSys*() qui renvoient respectivement la date et l'heure du serveur à l'instant de l'appel.
  A ce moment les données ne sont pas en base de données, elles ne sont stockées que dans le contexte de données serveur (moteur Webdev).

- On procède à l'ajout effectif dans le fichier grâce à la fonction *HAjoute*().
  Cette fonction renvoie un booléen indiquant si l'ajout a été effectué avec succès ou non.
  On teste cette valeur renvoyée et dans le cas où *HAjoute*() renvoie Faux, une erreur est survenue on l'affiche à l'écran (*info*() en Webdev = *alert*() en JS).

- On récupère la description de l'erreur grâce à la fonction HErreur() qui peut prendre un paramètre pour spécifier plus précisément l'information voulue.

Modification des données

Un exemple de modification de données dans un fichier, code de clic serveur sur un bouton :

```
HRecherche(Utilisateur,Pseudo,SAI_Pseudo)
SI HTrouve(Utilisateur) ALORS
	Utilisateur.Mail = "toto@supinfo.com"
	HModifie(Utilisateur)
FIN
```

- On initialise l'index de lecture sur l'enregistrement dont on souhaite modifier les données en recherchant et en vérifiant que l'on trouve bien un enregistrement correspondant.

- On modifie le contexte de données serveur, la rubrique mail est modifiée.
  A ce moment la modification n'est pas appliquée en base de données.

- On utilise *HModifie*() pour modifier effectivement la base de données.

Autres actions

Les fonctions HyperFile du W-Langage sont très nombreuses, elles permettent de couvrir les besoins d'accès aux données que ce soit en manipulation de données, définition ou gestion des connexions.

2.6.2. Utilisation des requêtes

Les requêtes sont bien plus intéressantes que les fonctions HyperFile car elles sont bien plus réutilisables entre les projets, on peut aussi profiter de leur optimisation automatique et surtout à l'utilisation elles sont plus sympathiques à utiliser.

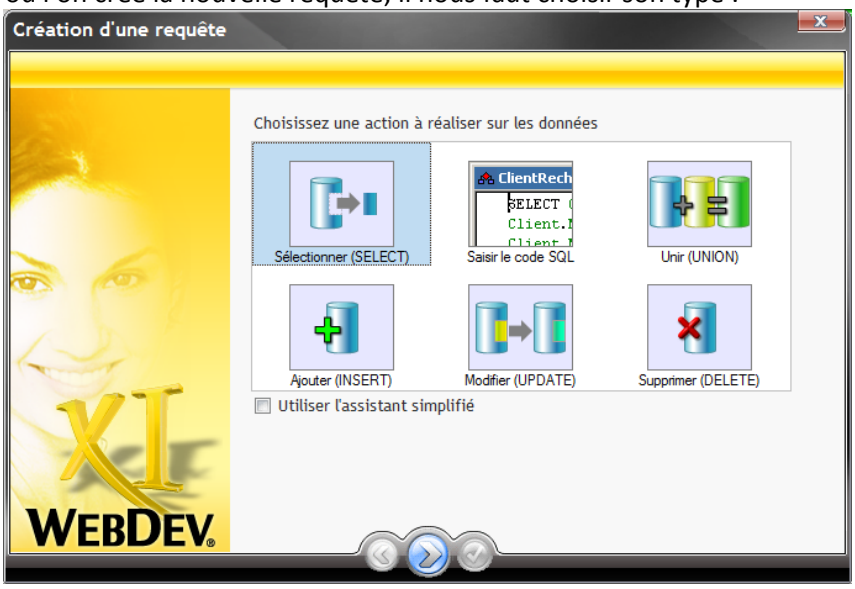
Pour créer un requête, on créé un nouvel élément de projet :

CTRL + N ou Fichier > Nouveau > Requête.

Requête de base

Par requête de base, on entend une requête qui récupère simplement un résultat filtré depuis la base de données.

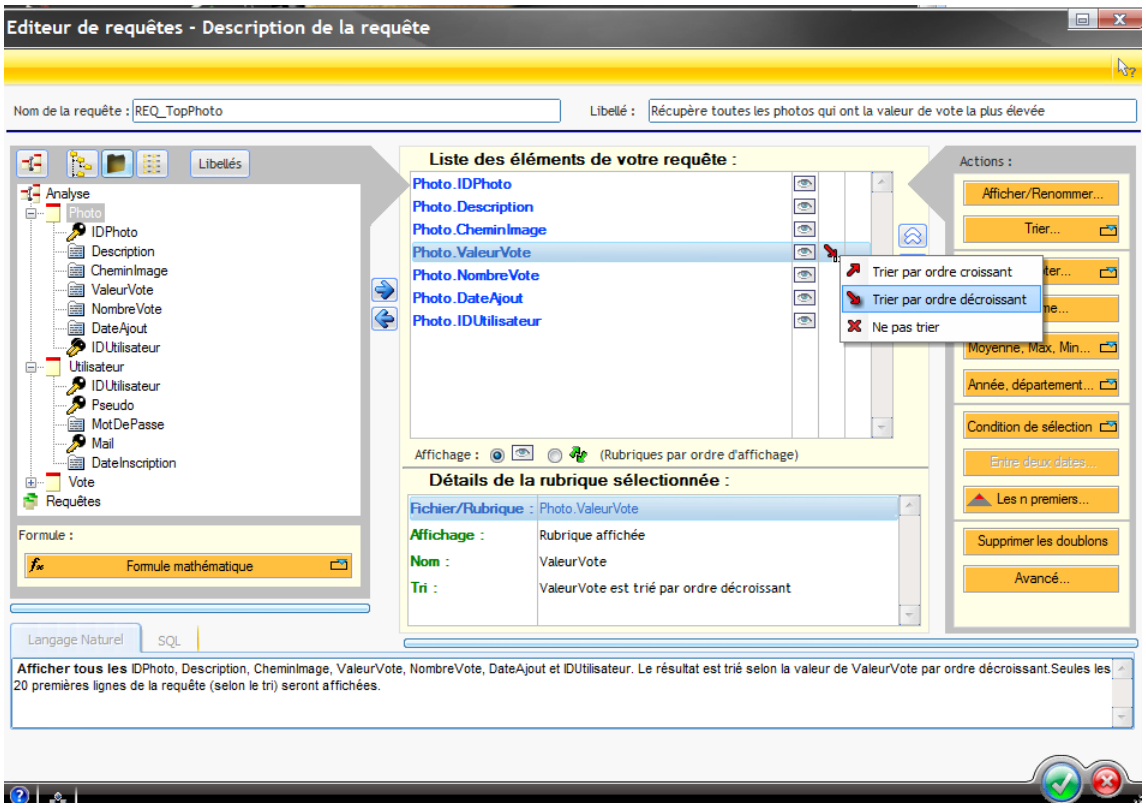
Au moment où l'on créé la nouvelle requête, il nous faut choisir son type :



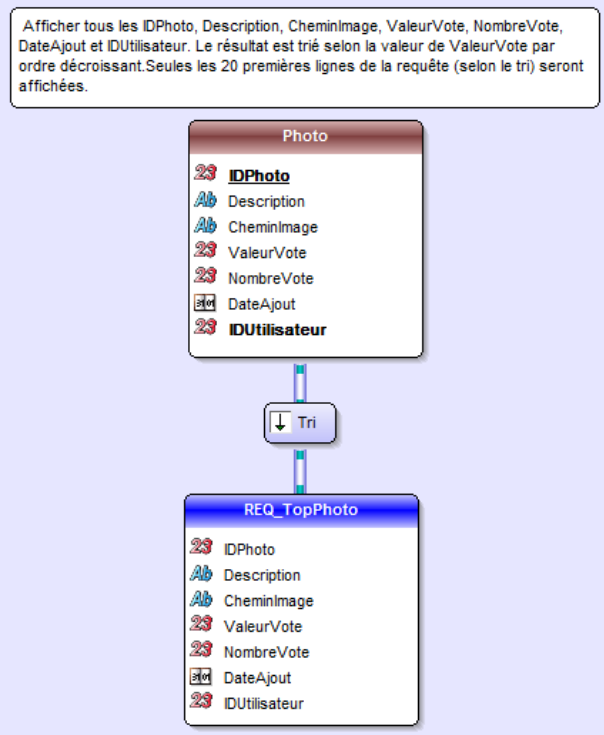
L'éditeur de requête apparaît. C'est un assistant pratique pour composer les requêtes de manière graphique en se concentrant sur les données que l'on souhaite intégrer plutôt que sur la forme de l'ordre à envoyer au moteur de base de données.

Son fonctionnement est relativement simple, sur la zone de gauche, on a un treeview représentant l'analyse et les requêtes déjà établies.
On sélectionne les rubriques que l'on souhaite ajouter en utilisant le bouton associé. Ces éléments sélectionnés sont affichés ensuite dans la liste centrale.

Il est alors possible d'indiquer des conditions de tri ou de filtre sur les rubriques sélectionnées. Il suffit de cliquer dans la liste sur la rubrique sur laquelle on souhaite appliquer le tri ou la condition. Une indication graphique nous permets de vérifier cela (et dans le cas d'un tri sur plusieurs paramètres, l'ordre d'application des tris successifs).



Une fois les éléments voulus sélectionnés ainsi que les différentes conditions établies, on peut avoir une représentation graphique de la requête.

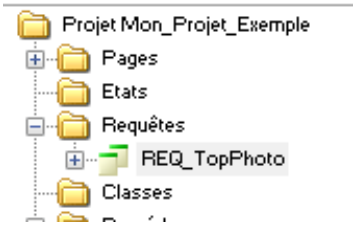


Dans ce cas il s'agit simplement d'une requête qui récupère les 20 premières photos qui ont la meilleure valeur de vote.

On peut bien entendu voir le code SQL associé à cette requête générée depuis l'éditeur graphique et le corriger, sur l'éditeur, on appuie sur F2 (raccourci pour voir le code des éléments), ce code est également affiché en bas de l'éditeur graphique de requête :

```
SELECT TOP 20
	Photo.IDPhoto AS IDPhoto,
	Photo.Description AS Description,
	Photo.CheminImage AS CheminImage,
	Photo.ValeurVote AS ValeurVote,
	Photo.NombreVote AS NombreVote,
	Photo.DateAjout AS DateAjout,
	Photo.IDUtilisateur AS IDUtilisateur
FROM
	Photo
ORDER BY
	ValeurVote DESC
```

Une fois enregistrée cette requête figure dans les éléments du projet et elle est regroupée dans le dossier « Requêtes » du projet.



Pour utiliser cette requête dans le code, dans le code de clic serveur d'un bouton par exemple :

```
SI HExécuteRequête(REQ_TopPhoto) ALORS
	HLitPremier(REQ_TopPhoto)

	SI HNbEnr(REQ_TopPhoto) > 0 ALORS
		TANTQUE PAS HEnDehors(REQ_TopPhoto)
			Info(REQ_TopPhoto.CheminImage)
			HLitSuivant(REQ_TopPhoto)
		FIN
	FIN
FIN
```

- Dans un premier temps on initialise la requête sur le serveur. Cela initialise le contexte de données serveur.
  Pour cela on utilise la fonction *HExecuteRequête*().
  Cette fonction prend en paramètre le nom de la requête à exécuter puis d'autres paramètres optionnels concernant l'accès au fichier et les paramètres éventuels de la requête.
  Cette fonction renvoie un booléen indiquant si la requête s'est correctement initialisée, on teste ainsi sur cette condition pour la suite du traitement.

- On positionne l'index de lecture sur le premier enregistrement des données récupérées à partir de cette requête.

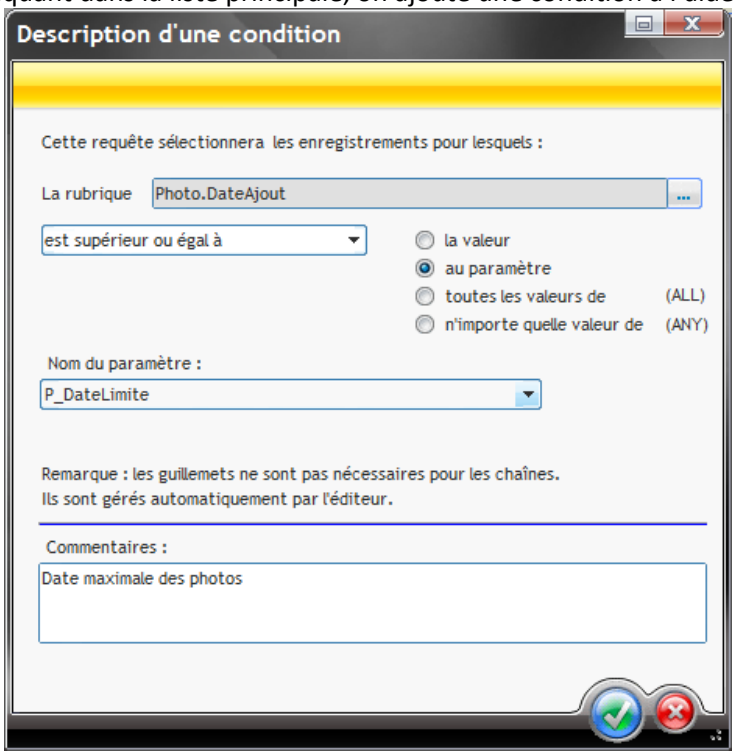
- On teste si la requête renvoie des résultats, et si c'est le cas on parcourt dans une boucle les résultats.

Requête évoluée et paramétrée

Dans le cas précédent, on a simplement sélectionné en effectuant un tri.
Webdev permet d'utiliser toutes les possibilités offertes par le langage SQL.
Ici la requête qui suit va permettre de récupérer les mêmes photos du Top 20 ainsi que les informations sur les utilisateurs à qui sont liées ces photos, enfin ces photos seront triées par valeur de vote et une condition sur les dates afin de ne pas récupérer celles plus anciennes qu'une date spécifiée en paramètre.

- On sélectionne tout d'abord les données que l'on souhaite, ainsi on récupère les données du fichier Photo, puis les données du fichier Utilisateur qui contient les informations sur l'auteur.

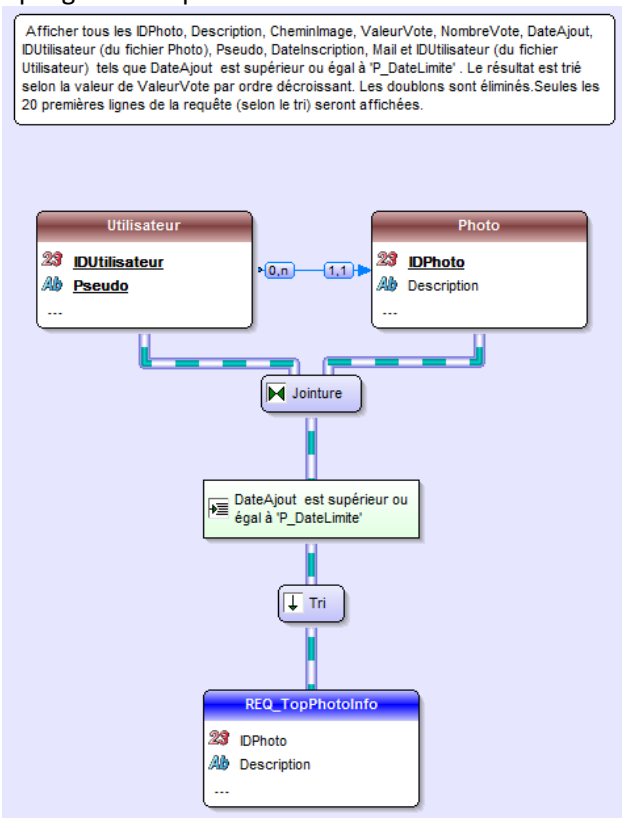
- On applique le tri sur la rubrique ValeurVote et on applique une condition sur la rubrique DateAjout : en cliquant dans la liste principale, on ajoute une condition à l'aide de l'assistant :



Il est très pratique de préfixer les noms des paramètres des requêtes notamment au moment de leur utilisation dans le code.

- On impose un nombre maximal de résultats à retourner en cliquant sur le bouton « n premiers » de l'éditeur de requête, et on supprime les éventuels doublons.

Voilà le résultat graphique généré depuis l'éditeur :



Le code SQL correspondant généré :

```
SELECT DISTINCT TOP 20
	Photo.IDPhoto AS IDPhoto,
	Photo.Description AS Description,
	Photo.CheminImage AS CheminImage,
	Photo.ValeurVote AS ValeurVote,
	Photo.NombreVote AS NombreVote,
	Photo.DateAjout AS DateAjout,
	Photo.IDUtilisateur AS IDUtilisateur,
	Utilisateur.Pseudo AS Pseudo_Ut,
	Utilisateur.DateInscription AS DateInscription,
	Utilisateur.Mail AS Mail,
	Utilisateur.IDUtilisateur AS IDUtilisateur_Ut
FROM
	Utilisateur,
	Photo
WHERE
		Utilisateur.IDUtilisateur = Photo.IDUtilisateur
	AND
	(
		-- Date maximale des photos
	Photo.DateAjout >= {P_DateLimite}
	)
ORDER BY
	ValeurVote DESC
```

Pour utiliser cette requête dans un code serveur, on peut utiliser cette notation :

```
dhMaDateLimite est une DateHeure

dhMaDateLimite..Mois = 3
dhMaDateLimite..Année = 2007
dhMaDateLimite..Jour = 12
dhMaDateLimite..PartieHeure = 0

REQ_TopPhotoInfo.P_DateLimite = dhMaDateLimite

SI PAS HExécuteRequête(REQ_TopPhotoInfo,hRequêteDéfaut) ALORS
       Info("ERREUR:"+HErreur(hErrPrincipal))
SINON
       HLitPremier(REQ_TopPhotoInfo)
       TANTQUE PAS HEnDehors(REQ_TopPhotoInfo)
              Info(REQ_TopPhotoInfo.CheminImage)
              HLitSuivant(REQ_TopPhotoInfo)
       FIN
FIN
```

- On déclare et initialise une variable de type DateHeure.
- On initialise le paramètre de la requête en utilisant la notation suivante : *NomRequete.NomParamètre* : ici l'intérêt de préfixer par « P_ » les paramètres des requêtes prend son sens puisque dès lors que l'on commence à saisir « P_ », on observe la liste des paramètres en auto complétion, bien moins difficile à identifier qu'un nom commun.
- On appelle la fonction *HExecuteRequête*() en testant la valeur du résultat de son initialisation.
- On positionne l'index de lecture sur le premier résultat et on lit les données dans une boucle.

Ces requêtes paramétrées peuvent être très élaborées, c'est notamment le cas dans le projet de stage, le préfixage sert véritablement, de même que les commentaires sur les paramètres.

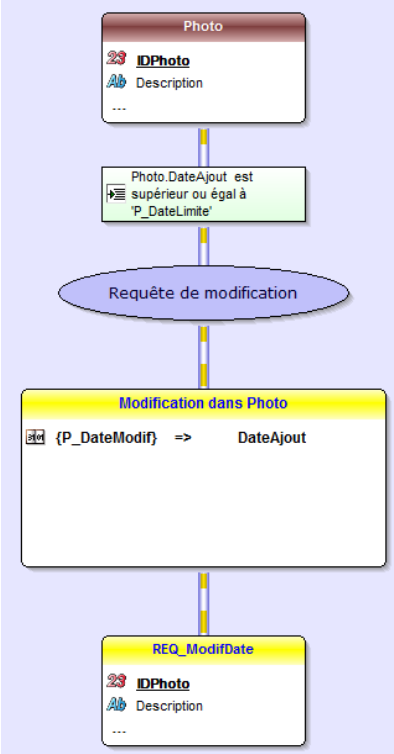
Requête de mise à jour de données

Après avoir sélectionné la création d'un nouvel élément de type requête via l'assistant dédié, on sélectionne le type de requête « Update » :

Il faut ensuite sélectionner le fichier sur lequel on souhaite travailler :

Ensuite, on observe la liste des rubriques du fichier, on peut alors imposer les conditions permettant de sélectionner les enregistrements qui devront être modifiés :

Après avoir nommé et décrit brièvement la requête, on peut valider et ainsi observer l'aperçu graphique de la requête :



Le code SQL généré correspondant :

```
UPDATE
	Photo
SET
	DateAjout = {P_DateModif}
WHERE
	-- Date maximale des photos
	Photo.DateAjout >= {P_DateLimite}
```

Toutes les possibilités d'édition et de composition de requête sont les mêmes que pour les requêtes de lecture.
Sur le même principe il existe des requêtes de suppression qui permettent de supprimer les champs que l'on souhaite selon les conditions indiquées.

2.6.3. Utilisation de requêtes SQL composées

Dans certains cas, le développeur a besoin de composer les requêtes selon son besoin. Indiquer sur quel fichier il travaille selon le choix d'un utilisateur, selon d'autres paramètres de l'application… Les cas possibles sont bien trop nombreux pour être présentés sous forme de requêtes établies.

Webdev permet bien entendu de composer ses propres requêtes comme on le ferait avec n'importe quel autre langage de programmation.

Pour cela on utilise une fonction du W-Langage qui permet d'exécuter une requête SQL composée sous forme de chaîne de caractères. Cette requête n'est pas vérifiée dans l'éditeur, c'est pourquoi les solutions précédentes sont conseillées. Les erreurs éventuelles arrivent pendant l'exécution d'où le besoin de les capturer comme il se doit pour éviter un crash de l'application ou du site.

Voici le code permettant de récupérer certaines rubriques spécifiées dans un tableau de chaînes dans un fichier lui aussi spécifié. Dans cet exemple, l'ensemble de la requête est composé :

```
//Paramètres externes
monOrdreSQL est une chaîne = "SELECT "
monFichierRequeté est une chaîne = "Photo"
sRubriqueTemp est une chaîne
maRequeteComposée est une chaîne

mesRubriquesSelectionnées est un tableau de 3 chaînes
TableauAjoute(mesRubriquesSelectionnées,"Description")
TableauAjoute(mesRubriquesSelectionnées,"CheminImage")
TableauAjoute(mesRubriquesSelectionnées,"IDUtilisateur")


//Composition de la requête
maRequeteComposée = monOrdreSQL
POUR TOUT ELEMENT sRubriqueTemp DE mesRubriquesSelectionnées
	SI sRubriqueTemp <> "" ALORS maRequeteComposée += monFichierRequeté + "." +
sRubriqueTemp + ","
FIN
maRequeteComposée = Gauche(maRequeteComposée,Taille(maRequeteComposée)-1)	//On
supprime la dernière virgule
maRequeteComposée += " FROM " + monFichierRequeté + ";"

//Utilisation de la requête composée
sdMA_REQ_Compose est une Source de Données

HExécuteRequêteSQL(sdMA_REQ_Compose,maRequeteComposée)
HLitPremier(sdMA_REQ_Compose)
SI HTrouve ALORS
	TANTQUE PAS HEnDehors
		Info(sdMA_REQ_Compose.Description)
	FIN
FIN
```

- Dans ce code, on initialise des variables qui simulent des variables de l'application ou lues ailleurs. Ceci est fait ainsi pour souligner l'aspect dynamique de la requête composée.

On définit l'ordre SQL, le fichier requêté, les rubriques sélectionnées.
Cette requête est relativement simple dans son utilité.

- On compose ensuite la requête SQL à partir des éléments.
  On récupère tout d'abord l'ordre SQL, puis les rubriques souhaitées dans une boucle sur les éléments du tableau.
  On indique la clause FROM et le fichier à requêter.

- On déclare une variable en tant que source de données. Cet élément sera celui qui permettra d'accéder aux données résultant de la requête.
  On initialise la requête à l'aide de la fonction *HExecuteRequeteSQL*() qui dans le cas présent prend en paramètre la source de données et le texte composant la requête SQL.
  La source de données s'utilise ensuite comme une requête ou un fichier pour parcourir les éléments.

On remarque ainsi que cette utilisation est plutôt fastidieuse. C'est pourtant encore relativement utilisé notamment pour des systèmes de fichiers multiples, pour conserver un historique des données sur des fichiers différents.

Bien sur on peut avoir des codes bien moins complexes et bien plus compréhensibles, mais c'est rarement le cas de l'utilisation de cette méthode.

```
ReqPhoto est une Source de Données
// Initialisation de la requête "Client"
HExécuteRequêteSQL(ReqPhoto,"SELECT Description FROM Photo")
```

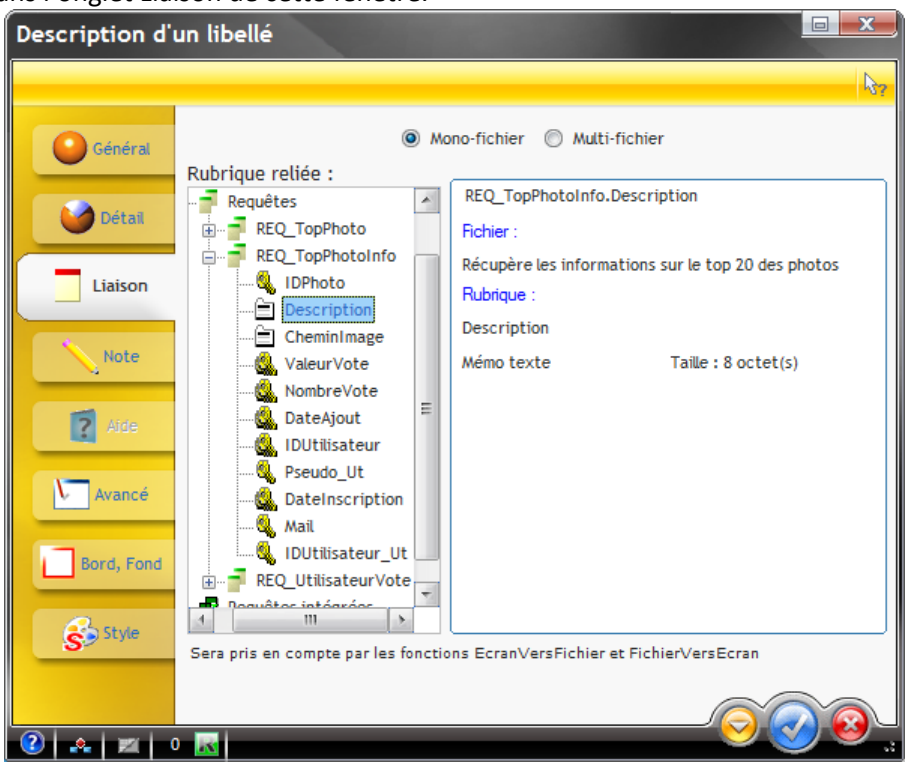
Cette utilisation est à éviter quand on le peut, ne serait-ce que pour le gain en temps, mais aussi et tout simplement pour la compréhension des autres personnes de l'équipe et la maintenabilité générale.

2.6.4. Liaison des champs aux données

La liaison des champs aux données correspond au binding direct de la base de données vers les champs et inversement.

Cette méthode est une solution permettant de faire gagner du temps dans le développement tout en améliorant la lisibilité du code.

Pour établir cette liaison des données sur les champs, dans la description du champ à lier, on configure dans l'onglet Liaison de cette fenêtre.



On sélectionne la rubrique à relier dans le treeview représentant les fichiers de l'analyse et les requêtes du projet ainsi que leurs rubriques.

Une fois cette liaison établie, il faut garder à l'idée que les données qui sont liées sont celles du contexte de données serveur, il est donc nécessaire, dans le cas d'une liaison sur une requête comme dans le cas d'une liaison sur un fichier, d'initialiser ce contexte.

Ici, on lie plusieurs champs aux rubriques de la requête REQ_TopPhotoInfo.
On souhaite ainsi afficher les informations de la première photo résultant de la requête.

Dans le cas de la requête précédente, le code se simplifie puisqu'on agit sur plusieurs champs en une seule instruction. L'association des données vers les champs s'effectue simplement.

```
dhMaDateLimite est une DateHeure

dhMaDateLimite..Mois = 3
dhMaDateLimite..Année = 2007
dhMaDateLimite..Jour = 12
dhMaDateLimite..PartieHeure = 0

REQ_TopPhotoInfo.P_DateLimite = dhMaDateLimite

SI PAS HExécuteRequête(REQ_TopPhotoInfo) ALORS
       Info("ERREUR:"+HErreur(hErrPrincipal))
SINON
       HLitPremier(REQ_TopPhotoInfo)
       FichierVersPage(MaPage)
FIN
```

Le code d'initialisation et de parcours de la requête ne change pas, seule l'association est simplifiée. En effet on utilise la fonction *FichierVersPage*() qui prend un paramètre optionnel correspondant à la page sur laquelle on souhaite effectuer le binding. Ici « MaPage » est un alias sur la page en cours, il n'est pas obligatoire dans ce cas d'utilisation.

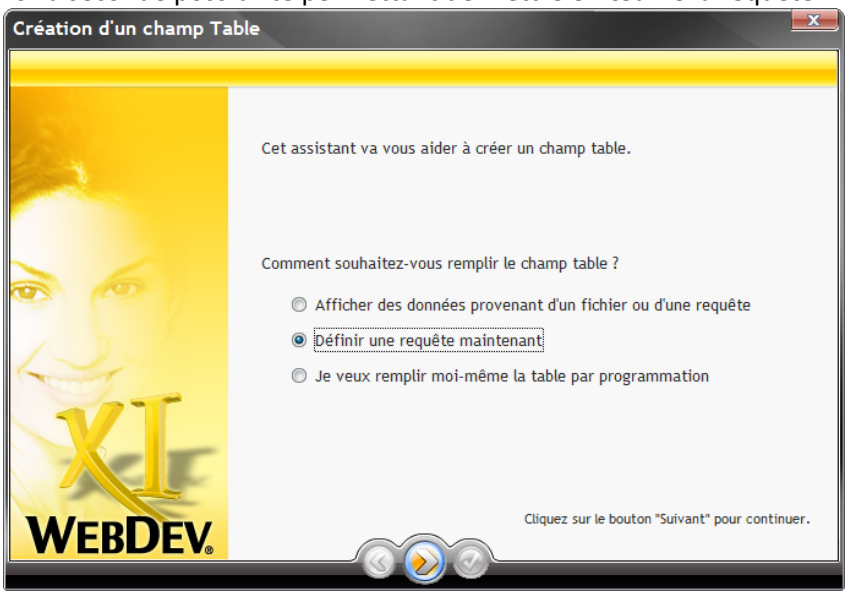
La liaison des champs peut bien sûr être combinée à une association manuelle.
Cette liaison s'applique également dans l'autre sens, de la page (des champs) vers la base de données, la fonction mise en œuvre est alors *PageVersFichier*().

Requête intégrée

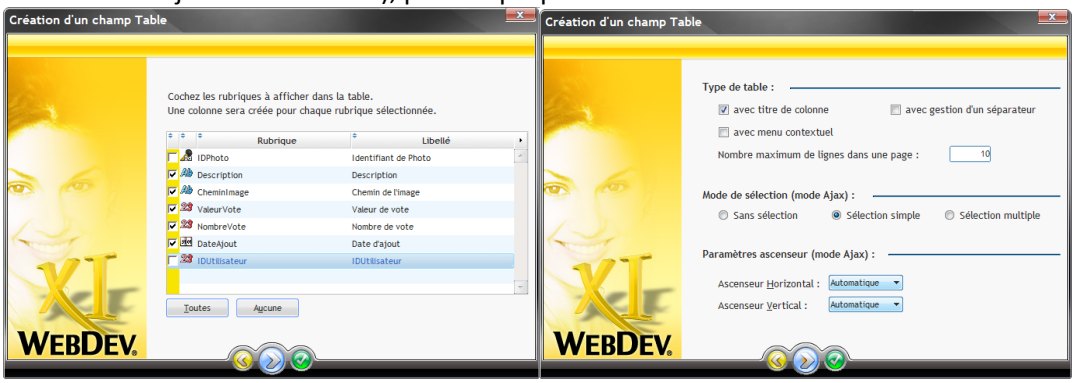
Lorsqu'on ajoute un champ à la page, on souhaite parfois l'initialiser de manière automatique sans avoir à composer une requête puisque ces données ne sont affichées qu'à travers ce champ.
Pas de réutilisation ailleurs, la solution est alors de composer une requête spécifique pour le champ : cette solution, c'est la requête intégrée.

Pour cela, on observe plusieurs solutions permettant de créer cette requête intégrée, la plus simple est de la créer en même temps que le champ, ici par exemple lors de la création d'un champ de type table, on propose de remplir le champ à l'aide des résultats provenant d'un fichier ou d'une requête, d'une requête que l'on souhaite définir maintenant (requête intégrée) et enfin de le remplir manuellement via code.

On choisit ainsi la seconde possibilité permettant de mettre en œuvre la requête intégrée :



L'assistant graphique d'édition des requêtes apparaît et on choisit les données que l'on souhaite exploiter, puis l'assistant de création du champ table nous invite à choisir les rubriques que l'on souhaite afficher dans la table, la rubrique de parcours (permettant de définir l'ordre d'ajout dans la table), puis les propriétés de la table et de mise en forme :



Une fois toutes les étapes validées, en exécution, la table est directement exploitable, le binding est simple, aucune ligne de code pour afficher ou exploiter les données.

La liaison des données est une fonctionnalité très puissante et utile qui fait gagner un temps précieux aux développeurs. Dans certains cas particuliers, on ne peut pas utiliser cette technique de liaison, il est alors utile de procéder de manière manuelle, mais lorsqu'on peut la mettre en œuvre, il ne faut absolument pas hésiter car cette liaison est également une source de rétro-modification : lorsqu'on modifie, par exemple, le format dans la base de données, le champ lié peut être modifié automatiquement si on le souhaite. C'est donc une valeur sûre en termes de maintenabilité.

2.7. Génération des états

Les états sont des rapports, des vues d'informations, des documents textuels ou graphiques, principalement destinés à être imprimés ou envoyés via un support différent d'une page Web.
On peut par exemple les utiliser pour envoyer des emails ou des courriers ou bien même directement générer des documents PDF ou sous d'autres formats.

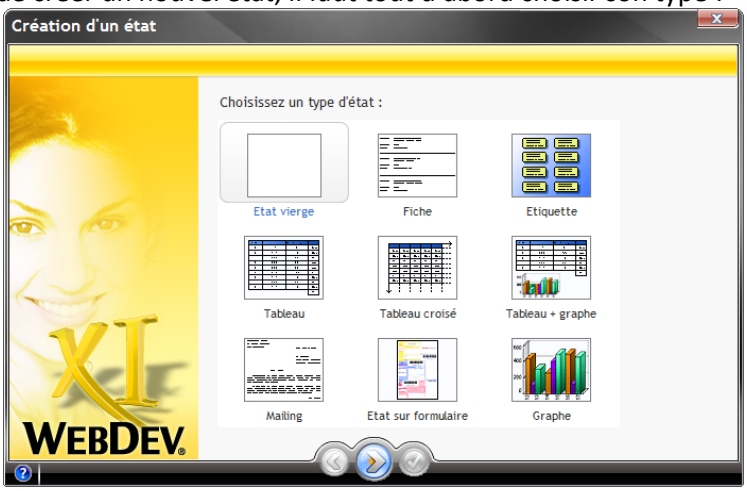
2.7.1. Etat simple

Par état de base, on entend un état qui ne contient pas de données particulières, il est établi de manière statique par le développeur pour afficher de l'information générale.

Pour créer un état, il suffit de procéder à la création d'un nouvel élément dans le projet :

Fichier > Nouveau > Etat.

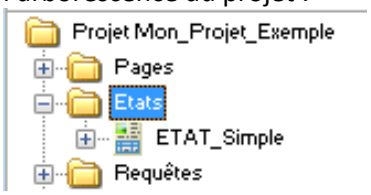
Après avoir choisi de créer un nouvel état, il faut tout d'abord choisir son type :



Le type d'état permet de créer les sections d'état, de configurer l'état en fonction du besoin et de créer des champs par défaut. Pour cet état de base, on choisit de ne pas utiliser de type particulier et on choisit donc le type d'état vierge.
L'assistant qui suit nous permet de définir la source de données, dans ce premier état, on choisit de ne pas utiliser de source de données.

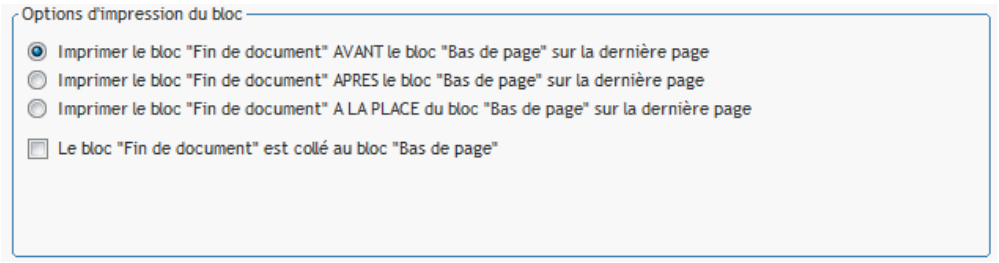
Reste ensuite à spécifier les informations de format et de style de l'état avant de saisir enfin le nom de l'état et une brève description.

Une fois les étapes de création validées, on peut voir l'aperçu de l'état dans l'éditeur d'état de Webdev, on enregistre l'état (par défaut l'extension est « .wde ») et on peut le voir ajouté dans l'arborescence du projet :



L'éditeur d'état nous indique que l'état est composé de différentes parties permettant d'assurer la mise en page de l'état final. La liste de ces différentes parties, appelées blocs dans l'éditeur, peut être éditée dans la description (Clic Droit > Description) de l'état.

Chacune de ces parties peuvent alors définir des propriétés et la forme du document correspondant à l'état final, dans la description de ces parties :



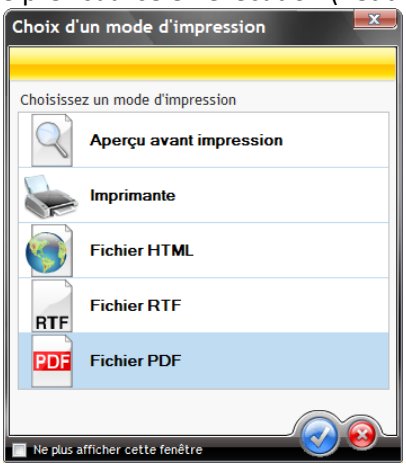
On peut ainsi paramétrer totalement l'état dans sa forme.

Pour ajouter des champs sur l'état, on procède comme avec les pages :

Barre de menu > Insertion > Nouveau champ…
Barre d'outils dédiée en utilisant un simple glisser/déposer

Les propriétés des champs peuvent être éditées via l'assistant de description des champs.

Une fois composé, l'état peut être prévisualisé en exécution (Petit Go) sous différentes formes :



Pour utiliser l'état en programmation, on a à sa disposition plusieurs fonctions, on peut même directement afficher cet état dans le navigateur :

```
NomFichier est une chaîne = "fichier.PDF"
iAperçu(iPDF, NomFichier)
iImprimeEtat(ETAT_Simple)
FichierAffiche(NomFichier, "application/pdf")
```

- On créé une chaîne correspondant au nom du fichier. Le fichier sera généré par défaut dans le répertoire WEB du site (répertoire racine du serveur pour les éléments du site).

- On initialise les paramètres d'impression en indiquant le type de fichier et le nom (et/ou chemin) du fichier généré à l'aide de la fonction *iApercu*().
  Cette fonction prend dans ce cas précis en paramètre le format du fichier (le type) et le chemin.
  Il existe de nombreux différents prototypes pour cette fonction permettant d'envoyer le document à une imprimante, via le fax et bien d'autres solutions.

- On imprime réellement l'état. L'état est initialisé et généré selon les paramètres indiqués. A ce moment le fichier PDF est véritablement généré.
  On utilise la fonction *iImprimeEtat*() qui prend en paramètre le nom de l'état à imprimer.

- La dernière ligne permet d'afficher le fichier dans le navigateur de l'utilisateur.
  On utilise la fonction *FichierAffiche*() qui prend en paramètre le chemin du fichier et le type MIME associé.

D'une manière générale, les fonctions préfixées par « iXXX » sont des fonctions qui traitent de l'impression ou la génération de fichier. Elles peuvent être utilisées pour travailler avec les états.

2.7.2. Etat paramétré

Dans certains cas, il peut être utile de passer des paramètres à un état pour le générer et ajouter des informations personnalisées.
Pour utiliser des paramètres au sein des états, il suffit de modifier le code d'ouverture de ce dernier et l'appel à la fonction *iImprimeEtat*() :

Dans le code d'ouverture de l'état :

```
PROCEDURE ETAT_Simple(gMonParametre1, gMonParametreOptionnel = "texte optionnel")
```

Le premier paramètre est obligatoire, le second est optionnel puisqu'il a une valeur par défaut.

Dans le code de lecture des données de l'état, pour utiliser ces paramètres :

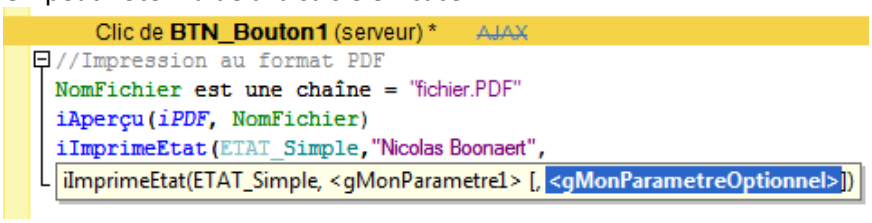
```
LIB_Nom = gMonParametre1 + "" + gMonParametreOptionnel
```

Les paramètres sont directement utilisables comme des variables.

Pour initialiser ces paramètres au moment de l'appel à *iImprimeEtat*(), dans le code d'un bouton de la page (code de clic serveur) :

```
//Impression au format PDF
NomFichier est une chaîne = "fichier.PDF"
iAperçu(iPDF, NomFichier)
iImprimeEtat(ETAT_Simple,"Nicolas Boonaert")
FichierAffiche(NomFichier, "application/pdf")
```

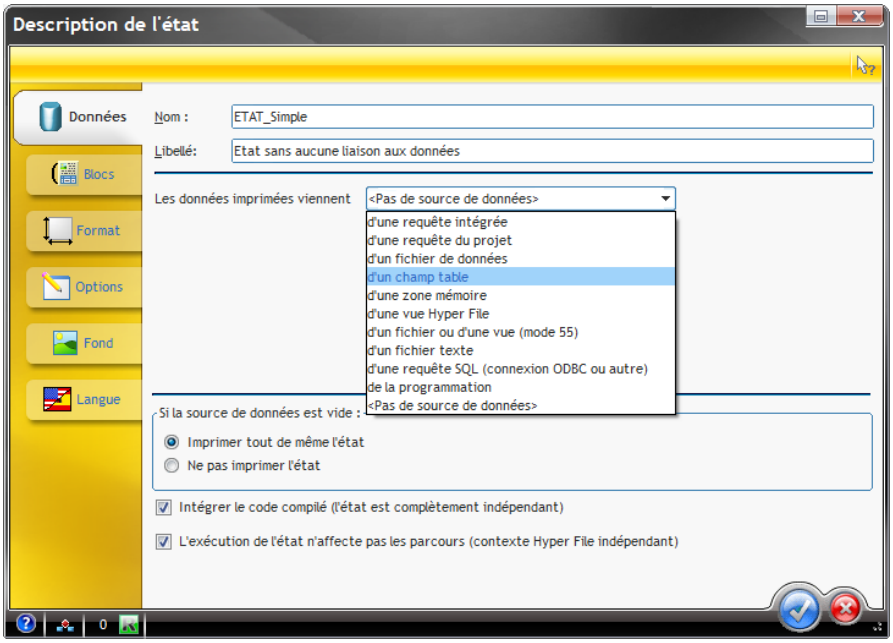
iImprimeEtat() prend les paramètres à la suite du nom de l'état à imprimer.
On peut noter l'aide à la saisie efficace :



2.7.3. Etat reposant sur la base de données

Parfois, on souhaite afficher des informations directement issues de la base de données sur un état, il est tout à fait possible de lier une source de données à des champs de cet état grâce à Webdev.

Dans le cas d'une création, il suffit de cliquer sur le type de source de données à associer, dans le cas présent, on choisit de modifier l'état de base, on entre ainsi dans la description de l'état et sur l'onglet Données.



Dans le cas mis en œuvre ici, on souhaite afficher les données provenant de la requête utilisée précédemment. Après avoir renseigné ce choix ainsi que le fichier à imprimer, nous allons voir les modifications que ça apporte.

Tout d'abord, puisqu'on souhaite afficher le contenu d'une requête, il est nécessaire d'initialiser la requête avant l'impression de l'état. Ainsi dans le code de génération/impression de l'état, on modifie le code :

```
dhMaDateLimite est une DateHeure

dhMaDateLimite..Mois = 3
dhMaDateLimite..Année = 2007
dhMaDateLimite..Jour = 12
dhMaDateLimite..PartieHeure = 0

//Impression au format PDF
NomFichier est une chaîne = "fichier.PDF"
iAperçu(iPDF, NomFichier)
iInitRequêteEtat(ETAT_Simple,dhMaDateLimite)
iImprimeEtat(ETAT_Simple,"Nicolas Boonaert")
FichierAffiche(NomFichier, "application/pdf")
```

- On ajoute l'appel à *iInitRequêteEtat*(), cette fonction prend en paramètre l'état sur lequel est établie la liaison avec la requête à initialiser, puis les paramètres éventuels de la requête.

Ensuite dans l'état, on ajoute par exemple un champ de type libellé, on souhaite afficher le contenu d'une rubrique, pour cela, on indique pour son contenu la forme suivante :

```
[%Description%]
```

D'une manière générale :

```
[%Rubrique%]
```

Lors de la lecture des données, cette notation est remplacée par la valeur du contexte de données.

2.8. Déploiement du site

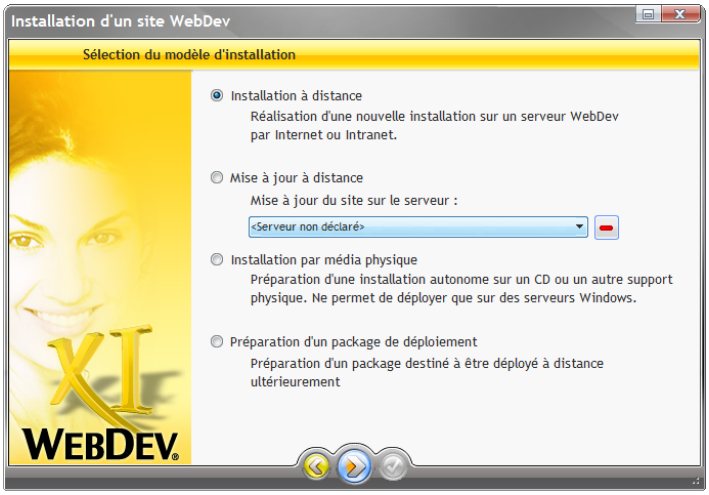
Le site réalisé pendant la période de stage a été déployé sur un système Windows Server 2003 utilisant IIS.

Pour déployer le site réalisé, PC Soft a une nouvelle fois simplifié cette étape en automatisant la procédure. Ainsi pour effectuer cette tâche :

Menu > Atelier > Créer la procédure d'installation.

L'assistant dédié apparaît et on peut alors sélectionner les éléments du projet à inclure dans la procédure d'installation ainsi que les langues.
On peut ensuite renseigner des informations qui seront utilisées dans la description du fichier d'installation telles que le nom de la société, la version, la description et les informations de copyright.

Après avoir validé les différentes étapes, l'assistant de création de fichier d'installation apparaît, on peut alors choisir le type d'installation que l'on souhaite utiliser : Installation à distance, mise à jour à distance, installation sur média physique (CD ou autre), Préparation d'un package de déploiement. Dans notre cas d'utilisation, on utilise l'installation via un media physique.



Après avoir renseigné ces informations, le fichier d'installation est généré dans le dossier de travail du projet, par défaut dans un sous dossier « INST » :

C:\Mes Sites\Mon_Projet_Exemple\INST

Pour installer le site, il suffit de double-cliquer sur le fichier généré sur le serveur, l'installation simplifiée arrête les services Internet du serveur (IIS ou Apache) et les redémarre juste après.

2.9. Mise à jour du site

2.9.1. Mise à jour de la base de données

Un projet évoluant en permanence en fonction des besoins, il arrive qu'on ait à modifier l'analyse (schéma de la base de données).
Lorsqu'on modifie l'analyse, il faut également modifier les fichiers de données utilisés par le site. En effet, pour éviter tout souci de corruption des fichiers de la base de données, il faut répercuter les modifications de la définition (analyse) vers le fichier de données.

Lorsqu'on est maître de projet, on peut modifier l'analyse.
Une fois l'ensemble des modifications effectuées, on peut régénérer l'analyse, son numéro de version est alors incrémenté, pour cela il faut cliquer sur l'icône correspondant :

Pour chaque génération, on peut voir une version d'analyse, les informations correspondantes sont stockées dans un dossier du projet : MonAnalyse.wd11/ANAXXXXX (ou XXXXX correspondant à la version).

Webdev nous informe, à travers un assistant de modification automatique, que certains fichiers ont été modifiés, cet assistant prend en compte les différentes connexions sur les bases de tests, de production et même les répliques de sauvegarde, le maître de projet peut alors choisir les bases dont les fichiers doivent être modifiés :

Puis il est possible de choisir sur le même principe les fichiers à modifier. Et il est également possible de gérer la sauvegarde avant modification effective des fichiers de données.
De cette manière, l'ensemble de l'opération de mise à jour des données est effectué avec sauvegarde, et en cas de problèmes dans cette étape critique, il est toujours possible de revenir à une version précédente fonctionnelle.

Lorsqu'on travaille en équipe, il arrive qu'on ne soit pas maître de projet ou qu'une autre personne ait eu à modifier l'analyse. En récupérant le projet ou la dernière version de l'analyse, on peut rencontrer des différences de versions entre l'analyse et les fichiers de données.

L'erreur se manifeste par ailleurs lors de l'accès aux données et donc en fonctionnement.

Pour mettre à jour une base de données qui ne l'a pas été (exemple d'une base locale non mis à jour par le maître de projet), on peut utiliser l'outil WDModFic du centre de maintenance.
Pour cela, à travers la barre de menu > Outils > WDOutil : Centre de maintenance.
Le centre de maintenance apparaît et regroupe l'ensemble des outils utiles de Webdev comme le comparateur de fichiers sources, la gestion de différence des analyses et bien d'autres outils de gestion de projet et de dépendances.

On sélectionne l'outil WDModFic qui est relativement simple à utiliser.
En effet il suffit de sélectionner l'analyse de référence pour la modification à travers l'assistant, puis la ou les bases à modifier ainsi que les fichiers.

Les fichiers de données sont mis à jour et sont alors en cohérence avec l'analyse sélectionnée, les données sont conservées s'il s'agit d'un ajout de champs ou même d'une modification de format, la modification s'effectue véritablement de manière automatique, on ne doit pas réintégrer les données manuellement par la suite.

3. Réflexion sur l'outil utilisé

Après plusieurs mois d'utilisation de l'outil Webdev, il est important d'avoir un œil critique posé sur l'outil.

Tout d'abord, Webdev améliore la productivité. C'est un fait indéniable et cela se remarque d'autant plus lorsque les développeurs et utilisateurs de Webdev maîtrisent l'outil. Cependant et il est important de le préciser, les projets mis en œuvres sont rarement aussi conséquents que d'autres réalisés sur des technologies plus répandues et bien plus robustes (J2EE, .Net…).

L'accès aux données, c'est-à-dire la facilité d'interaction avec la base est une force majeure de la suite de PC Soft. En effet Webdev, en simplifiant cet accès, apporte un gain de temps et d'efficacité tout en préservant une certaine flexibilité concernant cet outil et peut alors s'adapter à l'existant.

L'intégration des nouvelles technologies comme le support d'AJAX est également un réel avantage, l'évolution du produit se fait encore sentir sur ces points des nouvelles technologies, mais force est de constater que l'outil conserve simplicité et puissance.
Ces technologies évoluent au fil des versions cependant, il est très délicat de connaître les prochaines orientations des versions et on peut parfois être amené à fiabiliser un traitement, à créer un champ en passant par des astuces de développement, qui s'avèrent au final inutiles puisque la prochaine version les intègre simplement (glisser déposer ou propriétés particulières).

Le manque d'accès au code de sortie est également une limitation qui peut gêner mais qui respecte la philosophie de PC Soft sur le produit Webdev. En effet on se concentre davantage sur l'exploitation de ses éléments sans se contenter de ce qui est géré à plus bas niveau.
Webdev permet par exemple d'utiliser AJAX de manière simple : en un clic ou en configurant quelques informations sur le champ.

En résumé, Webdev est un outil de qualité, rencontrant parfois des soucis d'utilisation dans différents domaines et souffrant du manque d'une vraie communauté de développeurs.

L'outil évolue en permanence, et alors que la version 11 était utilisée pendant le stage et pour cet article, on vient de voir paraître la version 12 avec toujours plus de nouveautés.

4. Conclusion

4.1. Une base commune réutilisable

Le projet Spiléo s'appuie sur une base de données unique et commune à l'ensemble des sites réalisés.

Le site général continue d'évoluer au fil des versions tandis que plusieurs sites thématiques sont en cours de préparation en s'appuyant sur une base développée pour être réutilisée sans grand changement de code.

L'intégration de Virtual Earth au sein des sites thématiques a été revue pour simplifier la maintenabilité et l'ensemble du projet a été prévu pour pouvoir être facilement mis à jour tant sur la forme que pour les contenus.

4.2. Une ouverture vers d'autres technologies

En utilisant une base de données HyperFile Client/serveur, on assure une disponibilité renforcée et on profite de l'analyse établie au sein du projet.

Au sein des éditeurs de PCSoft à savoir Windev, Webdev et Windev Mobile, on utilise un seul et même projet. Il est alors possible d'utiliser la même analyse afin d'assurer une utilisation sur différentes plateformes et sous différentes formes.

Ainsi, sous Windev il est tout à fait possible de créer un web service qui pourra alors être interrogé à partir d'un client qu'importe le langage de développement pourvu de la possibilité de communication avec un Web Service. C'est d'ailleurs l'objet d'un des projets afin d'ouvrir la plateforme à d'autres technologies et diversifier les utilisations.