|  |  |
| --- | --- |
|  | **Master Informatique IHM Promotion 2011-2012** |

*Rapport de Stage*

Rapport de stage présenté   
par

Nicolas Reitz

En Vue de l’obtention du Master Informatique

Spécialité Interaction/Interface Homme Machine

|  |  |
| --- | --- |
| **Stage effectué A Atos** | 8 rue Lafayette 57000 Metz |

# Remerciement

# Sommaire

Remerciement 1

Sommaire 2

Table des illustrations 3

Abstract 5

Introduction 6

Présentation d’Atos 7

Sujet 8

Cycle de vie d’un projet 9

Le portail des Lorrains (PDL) 10

Présentation de Jahia 10

Développement 10

Syntaxe 11

Module 12

Template 12

Portlet 12

Les trois types de requêtes possibles sous Jahia 12

Travaux relatifs au Portail des Lorrains - version 1 13

FML 13

ReadSpeaker 17

Version 2.0 du Portail des Lorrains 18

Déroulement du développement de la version 2 18

Outils utilisés 18

Problématique du découpage de certains éléments 19

En-tête du site 20

Barre permanente 21

La cartographie des actions régionales 24

Recherche Synomia 26

Table des Annexes 27

Annexe A

Annexe i - Commentaire jour à jour A

Annexe ii - Autre B

Table des matières A

# Table des illustrations

Figure 1 - Cycle de vie d'un projet 8

Figure 2 - Schéma explicatif se basant sur une vue éclipse 10

Figure 3 - Détail d'un module 10

Figure 4 – Visualisation finale de la mise en forme sous deux colonnes 12

Figure 5 - Bouton Facebook visible par la région lorraine 13

Figure 6 - Bouton Facebook visible par un internaute lambda 13

Figure 7 - Schématisation de la problématique 14

Figure 8 – Comportement classique de la fenêtre de commentaire après clic sur le bouton « J'aime » 15

Figure 9 - Illustration de la méthode image 18

Figure 10 - Slider sans focus 19

Figure 11 - Slider avec focus 19

Figure 12 - Maquette de la barre 20

Figure 13 - Information visible avec JavaScript 20

Figure 14 - Information sans JavaScript 20

Figure 15 - Information avec JavaScript 21

Figure 16 - Information sans JavaScript 21

Figure 17 - Entrée par cible 22

Figure 18 - Espace privé 22

Figure 19 - Slider sans focus 22

Figure 20 - Slider avec focus sur un item 22

Figure 21 – Publicité non déployée 23

Figure 22 - Publicité déployée 23

Figure 23 – Ancien formulaire résultat attendu 25

Figure 24 – Ancien formulaire résultat obtenu sur certains navigateurs 25

Figure 26 – Nouveau formulaire résultat uniforme sur les principaux navigateurs 25

# Abstract

In order to obtain the Master in computer science with skilled in HCI (Human Computer Interaction), I did training in Atos Metz. This report contains a description of the subject, an overview of project management and different tasks done about de website of « Région Lorraine ». In these points, I talk about Jahia, the logic of this CMS and some components plugin in it: article can be voted, ReadSpeaker in the first version of Portail Des Lorrains. In the second part, I talk about the second version of the PDL and the decisions I had to take about the design and the problematic of accessibility (RGAA). On Annexe i - Commentaire jour à jour, there are some comments about the training from day to day.

# Introduction

Dans le cadre de l’obtention du diplôme de Master Informatique spécialité IHM, j’ai effectué un stage de 6 mois dans la SS2I Atos de l’agence de Metz. Ce rapport contient une description du sujet, un aperçu de la gestion de projet ainsi que les différentes tâches accomplies sur le site de la Région Lorraine. Dans ces points, j’aborde Jahia afin d’expliquer la logique de ce CMS, les nouveaux composants intégrés à Jahia : article votable, ReadSpeaker dans la première version du Portail des Lorrains. Dans une seconde partie, j’aborderai la deuxième version du site. En Annexe i - Commentaire jour à jour figure le suivi des tâches effectuées au jour le jour.

# Présentation d’Atos

# Sujet

# Cycle de vie d’un projet

Atos utilise la méthode de développement agile et met en application le principe de développement sur SVN. Ci-dessous figure un schéma représentant la démarche de gestion de projet.

Si besoin commentaire

**Chez le client**

Cahier des charges / Spécifications

Développement local + fiche de tests

Développement sur la plateforme de développement

Test sur la plateforme d’intégration

Test en pré production

Mise en production

Test sur la plateforme de développement

Fiche Mantis

Livraison

Figure 1 - Cycle de vie d'un projet

Le système de Mantis permet de faire le suivi de divers bogue, et nouvelles fonctionnalités souhaitées.

Les plateformes de développement et d’intégration se rapprochent des configurations du client.

Les fiches de tests reprennent les fonctionnalités qui doivent être testées en adéquation avec les spécifications du cahier des charges ou des spécifications. Dans un but d’optimisation seul est testé ce qui est impacté par le nouveau développement.

Une fois l’étape d’intégration effectuée, on établit un bon de commande, récapitulant différents éléments tels que les documents en relation, démarche d’installation, ce qu’il faut déployer, etc. puis on livre sous SVN.

# Le portail des Lorrains (PDL)

Le portail des Lorrains est un site web demandé par la région lorraine, il a été développé avec le CMS Jahia par Atos.

## Présentation de Jahia

Une formation de deux jours en Jahia6.5 m'a été dispensée afin d'avoir des bases pour le développement du PDL. Le PDL est un site web administré par la région lorraine. Ce site utilise le CMS Jahia, basé sur le langage java. Ce CMS autorise l’ajout de vue et la manipulation de la base de données Jahia. En effet, il intègre une base sur laquelle nous avons relativement peu de choses à faire pour l’administrer. Les opérations les plus délicates restent l’élaboration d’éventuelles requêtes SQL afin de réaliser des vues particulières. Jahia possède une sorte de méta-base de données afin d’administrer plusieurs sites internet en contenu, et en rôle et permission qui sont attribués à un utilisateur du service Jahia.

Ce CMS contrôle la saisie de certaines informations afin qu’elles soient conformes à certaines normes d’accessibilités tel que l’attribut « alt » à renseigner.

### Développement

Il est possible d’ajouter des éléments à Jahia. Ces éléments portent le nom de module, de template et de portlet. Le développement de ces composants se fait en Java, les vues sont gérées par des pages JSP. Jahia possède un mode édition et un mode « live ». Il est possible de faire un rendu « pauvre » pour l’édition et un contenu « riche » pour le mode « live » suivant le contexte d’utilisation du CMS. Le contenu de la page est à la discrétion du client qui construit ses pages avec l’aide l’éditeur mis à disposition par Jahia.

Nous utilisons Maven, afin de construire le fichier à déployer. C’est un outil pour la gestion et l'automatisation de production des projets Java.

Dans les parties suivantes, des explications sur les divers éléments abordés sont décrites.

### Syntaxe

On peut assimiler un module à un projet Maven. Un composant est une partie d’un module. Ce composant peut être caractérisé par différentes vues, pouvant posséder différents champs.



Les projets enfants

=

Modules

Fichier de préférence de déploiement

Fichier de configuration du projet parent

Figure 2 - Schéma explicatif se basant sur une vue éclipse



Vue du composant

Fichier de configuration du module

Composants du module

Figure 3 - Détail d'un module

### Module

Ajouter un module à Jahia, c’est ajouter une nouvelle famille de fonctionnalités. Un composant permet de réaliser le plus petit bloc de données identifiables. Par exemple, un article est un composant utilisant un titre, un contenu, des images ; un article votable est un composant nécessitant un titre, des descriptions courtes, éventuellement un lien et un bouton pour voter. Ces deux composants peuvent être réunis dans un même module (ici crlo – article)

### Template

Le template permet de définir un patron de vue pour une page, il définit donc les principales régions d’une page. Par exemple l’en-tête, le pied de page, la barre de menu pour une page classique. Ce patron est commun à toutes les pages du site. Le template est aussi la base pour la construction de nouvelle page.

### Portlet

Les portlets sont des applications placées dans un portail web. Elles utilisent le principe de servlets. Ces composants permettent de réaliser des services généralistes ou spécialisés tels que des annuaires, des moteurs de recherche, agenda, etc.

### Les trois types de requêtes possibles sous Jahia

#### Le SQL 2

Langage similaire à celui utilisé dans les bases de données. Les requêtes sont d’abord parsées et transformées en JQOM (Java Query Object Model) avant d’être exécutées. Du fait de son manque de maturité, il n’est pas optimisé pour les tâches complexes (jointure, etc.)

#### Le JQOM (Java Query Object Model)

Il s’agit d’une représentation en objet Java d’une requête. On peut construire ces requêtes à l’aide des bibliothèques de tags de recherche fournies par Jahia, ou bien directement à partir de code Java. Les requêtes SQL-2 et JQOM sont assez ressemblantes, sauf que les requêtes JQOM évitent l’étape de parsing et sont donc légèrement plus rapides.

#### Le XPATH

C’est le langage de requête le plus optimisé, mais il est aussi le plus compliqué à mettre en œuvre en raison de son écriture. L’écriture de ce type de requêtes ressemble au chemin d’une arborescence vers un fichier.

## Travaux relatifs au Portail des Lorrains - version 1

### FML

Le premier développement effectué consistait à créer un composant permettant aux visiteurs de voter pour un groupe de musique dans le cadre du festival des musiques lycéennes. Les contraintes données par le client étaient les suivantes :

L’affichage de « n » articles se fait sous forme de deux colonnes d’articles votables.



Composant article votable

Du module crlo - article

Figure 4 – Visualisation finale de la mise en forme sous deux colonnes

Le vote via Facebook avec différent affichage selon que l’internaute est un utilisateur lambda ou la région lorraine

|  |  |
| --- | --- |
| Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-04-10 à 23.29.18.png | Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-04-10 à 23.29.18.png |
| Figure 5 - Bouton Facebook visible par la région lorraine | Figure 6 - Bouton Facebook visible par un internaute lambda |

Le module doit être générique pour être utilisé dans différents contextes.

Il découle de ceci et de la maquette réalisée par le client, les 5 éléments suivants. Un titre, une image, deux brèves descriptions ainsi qu’un lien vers une vidéo externe. Le titre est un champ obligatoire, en effet il discrimine l’article votable.

#### Conceptions — différentes solutions possibles :

##### Solution 1

Cette solution consiste à déléguer l’affichage (placement sur la page) au module même via l’utilisation de propriété CSS. Elle est simple à mettre en œuvre, en effet elle nécessite uniquement de développer un nouveau composant « article votable » et d’utiliser les propriétés CSS de sorte qu’un article occupe 50 pour cent de l’espace disponible. Ainsi le second composant peut venir se placer directement à côté. L’inconvénient de cette méthode est de détecter, dans le module courant, si celui-ci est le dernier élément. En effet, dans le cas d’un nombre impair d’éléments le prochain contenu est susceptible de se placer au mauvais endroit. De plus, il devient délicat de gérer ce problème si l’on souhaite faire des regroupements d’articles dans différentes catégories.

Cette solution, bien que bonne sur le principe, semble tout de même souffrir de quelques inconvénients qui empêchent la réutilisabilité dans différentes configurations.



Figure 7 - Schématisation de la problématique

##### Solution 2

Cette solution consiste à déléguer l’affichage des articles à un module tiers. Cette méthode nécessite le développement d’un nouveau composant « article votable », mais aussi de développer une vue « 2colonnes » qui aurait pour rôle de structurer les articles en colonne et d’en afficher qu’un certain nombre. Cette solution permet de diminuer l’interaction avec les CSS et par la même occasion l’autoplacement d’un module comme expliqué ci-dessus. Cette méthode à l’avantage d’être itérative au contraire de la première solution, et donc de parfaitement maîtriser le positionnement du contenu.

##### Solution retenue

Afin de pouvoir contrôler correctement l’affichage, il a été choisi d’appliquer la deuxième solution.

#### Problèmes rencontrés

Le souci principal de ce développement a été le système de vote via Facebook. En effet, la demande du client détourne un peu l’utilisation du système de « J’aime ». La solution retenue pour masquer le compteur est de rajouter un « bandeau », de couleur identique au fond, au-dessus du compteur afin de le masquer. La seconde solution consistait à contraindre l’affichage au seul bouton. Cette méthode posait problème, car l’affichage de la zone de commentaire empiétait sur le bouton Facebook, ceci se produisant sur certains navigateurs (Internet Explorer et Firefox).



Figure 8 – Comportement classique de la fenêtre de commentaire après clic sur le bouton « J'aime »

Le second souci est de détecter qui est connecté afin de déterminer si le compteur doit être affiché ou non. Dans le contexte d’utilisation du site, il n’y a pas de notion de compte utilisateur, on ne peut donc pas se baser sur l’identifiant de l’utilisateur courant. Néanmoins, Jahia utilise un système de gestion de rôles et de permissions dans l’administration de ses sites. On ajoute donc une permission à un rôle qui sera attribué à un utilisateur connecté à Jahia autre que celui attribué au visiteur lambda.

#### Les remarques du client

Le client a demandé l’ajout d’une fonctionnalité qui a échappé à notre attention. En effet, la demande d’afficher l’intégralité du texte masqué par les points de suspension « … » dans une bulle d’information a été faite. La solution de base consiste à utiliser l’attribut « title » néanmoins, cette solution pose problème, car le « title » est prévu pour se masquer au bout d’un certain temps. La balise « abbr » a donc été utilisée, elle présente les mêmes caractéristiques que l’attribut « title » à l’exception de ne pas masquer le texte au bout d’un délai delta.

### ReadSpeaker

Dans l’optique de rendre accessible son site, la région lorraine a demandé d’intégrer ReadSpeaker à certains modules. ReadSpeaker est un logiciel de synthèse vocale qui convertit les textes HTML vers le format audio.

## Version 2.0 du Portail des Lorrains

La région lorraine a décidé de faire une mise à jour majeure de son site, publié en janvier 2012, afin de mieux répondre aux nouveaux besoins de la région. Cette mise à jour porte aussi bien sur le design et la charte graphique du site que sur les différents modules existants. De plus ce site étant commandé par une collectivité territoriale française, il doit respecter les normes d’accessibilités. La région demande le niveau AA du RGAA (Référentiel Général d'Accessibilité pour les Administrations).

Une équipe de trois développeurs est prévue pour le développement de la version 2. Loïc Gangloff, Alexandre Lebeau ainsi que le rédacteur de ce rapport[[1]](#footnote-1).

Afin de simplifier toute manipulation JavaScript, le framework JQuery est utilisé. La région lorraine a fourni une charte de graphique afin de nous aider dans le développement. Une des contraintes imposées par le client est que les fonctionnalités soient opérationnelles sous Internet Explorer[[2]](#footnote-2) et Firefox[[3]](#footnote-3).

### Déroulement du développement de la version 2

Dans un premier temps, à partir d’un document du client (CCTP), nous avons fait du prototypage afin de vérifier la faisabilité en ce qui concerne les fonctionnalités. Une fois obtenu un résultat satisfaisant, les prototypages ont continué sous Jahia, afin de s’assurer que la solution retenue soit applicable au CMS. Lorsque le projet a pu réellement commencer, nous avons rédigé la rédaction de divers documents concernant la mise à jour du PDL. Ces documents servent à assurer le suivi des mises à jour du projet et nous assurent de notre compréhension vis-à-vis de la demande du client. Dans un troisième temps, nous avons réalisé une maquette incluant les éléments de la charte graphique, afin que le client ait un aperçu du résultat, sur les jeux de couleurs (incluant la transparence), ainsi que la mise en page. La réalisation de cette maquette HTML a permis de faciliter l’intégration dans Jahia en évitant de déployer les modules concernés par la modification de trop nombreuses fois. Enfin, nous avons intégré dans Jahia l’ensemble des demandes.

### Outils utilisés

Afin de pouvoir manipuler la charte graphique, au format PSD[[4]](#footnote-4), nous avons utilisé dans un premier temps le logiciel Gimp. Dans la constatation des différences de visualisation entre Gimp et Photoshop (notamment dans la nuance de couleur), nous sommes passés sous Photoshop : version d’évaluation.

Les développements, réalisés sous Eclipse, ainsi que les documents relatifs au projet sont stockés sous subversion sur les serveurs de la région lorraine.

### Problématique du découpage de certains éléments

La maquette à une résolution de 2000x3000, il a donc fallu faire des redimensionnements afin d’avoir un visuel esthétiquement correcte pour des résolutions d’écran plus communes.

Dans la charte graphique, certains éléments, tels que des fonds pour menu déroulant, doivent avoir des bords arrondis en plus d’une transparence. Il existe différentes méthodes afin de faire ceci. Elles ne sont pas toutes compatibles avec les différents navigateurs existants.

Afin d’avoir un comportement correct sur le plus grand nombre de navigateurs, on peut utiliser diverses propriétés du CSS3 qui permettent d’afficher les arrondis ; ces propriétés sont malheureusement inefficaces sous Internet Explorer de même, le fond ne peut être remplacé par une couleur de fond de type RGBA[[5]](#footnote-5). Ces diverses observations impliquent l’utilisation d’une image dont le format autorise la transparence, par exemple le PNG.

Une autre contrainte implicite est que le fond soit adaptable à toute longueur de texte, pour être utilisable dans le plus grand nombre de contextes. Cette contrainte implique que l’on ne peut utiliser plusieurs composantes d’une image pour faire ce fond, en effet cette méthode nécessite des valeurs fixes pour avoir une imbrication correcte de toutes les composantes de l’image.

C:\Users\Nicolas\Desktop\Rapport Stage\image\Bouton de Base.png

Figure 9 - Illustration de la méthode image

Une autre solution consiste à n’utiliser qu’un pixel et à le répéter sur tous les points d’abscisse et d’ordonnée de la zone. Cette solution permet de gérer correctement l’extension du fond. Le dernier problème est la gestion des arrondis. Deux solutions sont possibles. La première est de mettre une image ou un fond d'écran supplémentaire dans chaque coin de la zone ; le problème inhérent à cette méthode est que l’image de fond des coins soit identique à la couleur de fond environnant. Une autre solution consiste à créer un style pour Internet Explorer, on peut inclure ce nouveau style en utilisant la propriété CSS « behavior », en lui précisant un fichier HTC[[6]](#footnote-6). Cette méthode permet ainsi d’obtenir le même effet sous tous les navigateurs, dont Internet Explorer.

### En-tête du site

#### Logo

Ce logo a été conçu de sorte qu’il respecte le RGAA en plus de favoriser le référencement. La solution est lisible correctement par les lecteurs d’écrans (Lynx, VoiceOver).

#### Accroche

D’après les spécifications envoyées par le client cette phrase est placée sous le titre du site, et est limitée en taille à une ligne. Cette notion subjective est dépendante des « dimensions » du site. En raison de son positionnement, une zone a été prévue sur le template afin qu’il n’y ait pas de problème de placement du composant dans l’interface de Jahia.

#### Slider partenaire

Ce slider permet de mettre en évidence les 5 sites satellites de la Région Lorraine (« Entreprenez en Lorraine », « mylorraine.fr », « Le site officiel du tourisme en Lorraine », « Eureka Lorraine », « INFFOLOR »). Un existant étant déjà présent, quelques modifications ont été demandées.

|  |  |
| --- | --- |
|  |  |
| Figure 10 - Slider sans focus | Figure 11 - Slider avec focus |

Il y a différentes interactions au niveau de ce slider. Le premier est une animation sur l’ensemble des items, à l’état « zéro » la représentation est sous forme pyramidale, à l’état « un » tous les items sont au même niveau. De plus, au survol d’un des items présents sur le bandeau, une zone est mise à jour avec le lien survolé.

##### Le framework ContentFlow

L’existant se base sur ce framework dont il est possible de modifier la vue de façon basique. Du fait qu’il ne contient pas tous les événements existant en JavaScript, il a fallu lui rajouter une couche indépendante afin de créer l’animation lors de la prise du focus.

##### « Partenariat » avec le slider de la barre permanente

Dans la partie suivante figure la description du slider « jumeau ». Ces deux sliders partagent dans ce cas le même modèle (liens vers les sites satellites), car ils ont le même rôle, celui de raccourci vers les autres sites de la Région Lorraine.

##### Respect du RGAA

Afin de respecter une des consignes du RGAA, la balise « noscript » est utilisée afin de pallier la désactivation du JavaScript.

### Barre permanente

Dans le cahier des charges figurent certaines exigences telles que la présence de menus déroulants nommés « espace privé » et « entrée par cible ». Ces menus sont présents dans une barre qui a une position fixe à l’écran. Elle est semblable à une barre des tâches et offre des accès rapides à certaines parties du site (page de recherches, sites satellites, liens externes et internes, etc.).

Figure 12 - Maquette de la barre

La barre est un des composants de la charte qu’il a fallu redimensionner afin qu’elle soit entièrement visible.

#### Problématique du respect du RGAA

##### Par rapport au JavaScript

L’un des critères du RGAA consiste à avoir les mêmes informations avec et sans le JavaScript en terme de texte. On peut aborder le JavaScript sous plusieurs angles.

###### « Développement parallèle »

On peut développer en parallèle le JavaScript et le CSS, dans ce cas le risque est pris de perdre des informations si le JavaScript est désactivé. Car on peut avoir tendance à fixer le CSS en fonction du premier état de l’animation. Voici des aperçus tirés d’un prototype.

|  |  |
| --- | --- |
| Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-05-05 à 03.58.45.png | Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-05-05 à 03.58.45.png |
| Figure 13 - Information visible avec JavaScript | Figure 14 - Information sans JavaScript |

###### « Développement procédural »

Une autre solution consiste à développer d’abord le CSS en s’arrangeant pour que toutes les informations soient correctement agencées et lisibles et seulement ensuite introduire le JavaScript. À ce niveau le JavaScript à un rôle de restructuration du contenu en intervenant au niveau du CSS (style) sur différentes propriétés (hauteur, largeur, couleur, etc.).

|  |  |
| --- | --- |
| Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-05-05 à 04.09.43.png | Macintosh HD:Users:Nico:Desktop:Capture d’écran 2012-05-05 à 04.16.21.png |
| Figure 15 - Information avec JavaScript | Figure 16 - Information sans JavaScript |

##### Par rapport au cahier des charges

Les fonctionnalités développées en JavaScript doivent donc être conçues soigneusement afin que les informations puissent être accessibles si le script est désactivé. En effet, si elles sont mal conçues elles seront invisibles à l’écran. En plus de poser des problèmes d’accessibilités, une mauvaise conception peut entraîner un mauvais référencement pour les moteurs de recherches tels que Google.

Afin d’éviter tout problème de ce genre, la solution adoptée consiste à utiliser un développement procédural. Dans l’application de ce principe à un menu déroulant, la mise en « menu déroulant » est entièrement faite en JavaScript (jQuery) ainsi s’il est désactivé, le CSS permettra à toutes les informations d’être visible sur la page. De cette manière, le CSS sera chargé de faire la mise en forme sans animation, et le JavaScript mettra en forme le contenu avec les animations qui conviennent.

#### Généralité sur l’« entrée par cible » et l’« espace privé »

Les menus figurants dans la barre permanente fixe ont l’obligation de s’ouvrir vers le haut, du fait même que cette barre a une position absolue en bas de l’écran.

Ces deux menus sont liés par un lien d’exclusion mutuelle. C'est-à-dire que si l’un des menus est visible, le second ne doit pas l’être. Afin de gérer au mieux les similitudes de ces deux menus et la relation qui les lie, un framework a été développé en tant qu’extension de jQuery, jQueryMenu.

Il est chargé de deux missions. La première est d’initialiser le composant. L’initialisation consiste à « plier le menu », puis attribuer la fonction d’animation. La première étape, dans le cas où le JavaScript est désactivé est ignorée ce qui permet de garder un rendu des différents éléments composant le menu.

##### Entrée par cible

Ce composant contient deux éléments. Le premier est le menu déroulant. Le second est un libellé qui sert de raccourci vers le dernier lien activé. Ce lien est stocké dans un cookie qui est renseigné lors du clic sur un des liens composant le menu. Afin d’inciter l’utilisation de ce menu, il est par défaut ouvert tant que le cookie est vide. Dans ce cas, il s’agit d’ajouter une nouvelle vue au composant existant.

Figure 17 - Entrée par cible

##### Espace privé

Ce menu est un simple menu déroulant. Bien qu’il fasse partie de l’existant, sa nouvelle position, dans la barre permanente (originellement dans l’en-tête), oblige à refaire le menu afin qu’il s’ouvre dans le bon sens.

Figure 18 - Espace privé

#### Slider partenaire

Au vu de l’espace alloué à ce slider (taille de la barre), le comportement est différent. Pour celui-ci, seul l’item courant est animé. Il grandit sensiblement par rapport aux autres. De la même manière que pour le premier slider, un événement est ajouté en guise de couche supplémentaire au framework, afin de créer cette animation.

|  |  |
| --- | --- |
|  |  |
| Figure 19 - Slider sans focus | Figure 20 - Slider avec focus sur un item |

De plus, les données de ce slider sont identiques à celles figurant dans la têtière. Afin d’éviter la redondance d’information, un composant particulier a été développé. Celui-ci permet de sélectionner un autre élément figurant dans la page et de récupérer le modèle associé à cet élément ; ceci est rendu possible grâce à l’héritage d’un composant Jahia. Ce système permet de manipuler, entre autres la vue, tout en partageant le même modèle que le composant précédemment lié. Ainsi, modifier l’élément de la têtière, modifie également celui figurant dans la barre permanente.

#### Outil publicité

Cet outil a les mêmes contraintes que les menus, à savoir s’ouvrir vers le haut. Il met donc aussi en œuvre le plug-in jQueryMenu. De plus, une période de validité contraint l’affichage de la publicité à cinq secondes, elle reste néanmoins accessible via un bouton dans le cas où la publicité est hors date de validité.

|  |  |
| --- | --- |
|  |  |
| Figure 21 – Publicité non déployée | Figure 22 - Publicité déployée |

### La cartographie des actions régionales

#### L’existant

La cartographie est un module qui permet de représenter sous forme de carte la répartition des diverses actions engagées par la région. Ce module est accompagné d’un formulaire de recherche servant de filtre qui permet de restreindre les résultats selon les années, et divers axes politiques.

#### Le souhait de la région

La région souhaite ajouter un élément dans le formulaire afin de pouvoir effectuer une recherche par mots clefs. Ces mots clefs permettront de restreindre les résultats sur le libellé, le territoire (ville), le département, ainsi que le nom du fichier PDF associé, contenant l’action entreprise. Cette mise à jour s’accompagne du remplacement de la carte existante ainsi que de la vue du module.

##### Le formulaire

Certaines modifications ont été apportées afin d’améliorer l’accessibilité. En effet, certains éléments du formulaire ont été détournés de leur fonction première. Par exemple, le bouton validant les données avait une méthode « onclick »[[7]](#footnote-7) associée, il n’était donc pas dépendant du formulaire. Ceci ayant pour conséquence de poster des données pouvant être erronées ou encore désactiver les fonctionnalités inhérentes au formulaire[[8]](#footnote-8) à savoir l’envoie du formulaire sans passer par le JavaScript.

L’action sur ce bouton a donc été déplacée dans l’appel à la validation du formulaire par le biais de l’attribut « onsubmit » de la balise « form ». Cependant, la validation de ce formulaire entraîne aussi le rechargement de la page, ce qui n’est pas souhaité, car des actions en AJAX sont associées à ce formulaire. La particularité du « onsubmit » est d’attendre le retour d’un booléen qui indique si l’envoi doit être exécuté ou non. Ce booléen vaut « vraie » par défaut, ce qui a pour conséquence de poster le formulaire. En rajoutant un « return false », le formulaire n’est pas posté, et l’AJAX est exécuté.

En procédant de cette manière, on ouvre de nouvelles possibilités à ce formulaire, par exemple l’envoie des données par l’appui sur la touche « Entrée ».

De plus, le formulaire a été restructuré, des balises « label », « fieldset », « legend » ont été rajoutées afin d’améliorer la structure du formulaire et son accessibilité.

Le travail sur ce formulaire permet de définir un style par défaut pour les éléments de formulaire qui seront utilisés sur le portail.

###### La balise « label »

Cette balise permet de décrire un champ du formulaire, elle est indispensable à la compréhension d’un formulaire pour de nombreuses personnes employant des méthodes de navigation alternatives. Sans elle, ces personnes seront dans l'impossibilité de comprendre le rôle de chaque champ du formulaire, et donc de le remplir.

De plus, l’un des avantages apportés par cette balise est de pouvoir offrir un accès plus rapide au champ qu’elle qualifie. Par exemple, un clic sur le label permettra de mettre le focus sur une zone de saisie ou de cocher un bouton « radio » ou « checkbox ».

###### Les balises « fieldset » et « legend »

La première balise sert à définir une zone particulière d’un formulaire, elle sert à regrouper les champs d’un formulaire par thème. La seconde balise permet de donner un titre à la zone du défini par « fieldset ».

##### Modification de la requête SQL

Cette requête est écrite avec le langage JQOM, c’est un langage qui permet d’écrire une requête en utilisant des tags. La modification de cette requête a été délicate pour la partie concernant la recherche dans le nom du fichier joint à l’action régionale. En effet, le système de gestion de données de Jahia permet de définir des weakreferences[[9]](#footnote-9). Ce système empêche d’accéder directement aux données relatives à l’objet possédant ces attributs. Une fois ce système de gestion assimilé, une parade a été trouvée ; celle de passer par une requête récupérant la liste des fichiers correspondants aux mots clefs, puis d’injecter ces résultats dans la requête principale.

##### La nouvelle carte

La cartographie possède certaines animations, en effet au survol d’une région, la région courante est mise en surbrillance, ce système est géré via les balises « map » et « area » qui permettent de délimiter diverses zones. Au survol d’une région, une zone flottante s’ouvre avec les actions dépendantes du territoire sélectionné.

La nouvelle carte étant plus détaillée, il faut remplacer la carte actuellement utilisée. Afin de minimiser le changement à effectuer sur les animations, la nouvelle est redimensionnée en suivant la carte utilisée dans la première version de portail.

### Recherche Synomia

De la même manière, une mise à jour de la vue est demandée. Cette mise à jour permet de corriger un défaut d’agencement sur plusieurs navigateurs. À cette occasion, un découpage plus minutieux a été fait ; l’image du bouton a été extraite pour être placée sur le bouton de validation de formulaire, la forme de la zone de saisie est stylisée grâce au CSS.

|  |  |
| --- | --- |
|  |  |
| Figure 23 – Ancien formulaire résultat attendu | Figure 24 – Ancien formulaire résultat obtenu sur certains navigateurs |
|  | |
| Figure 26 – Nouveau formulaire résultat uniforme sur les principaux navigateurs[[10]](#footnote-10) | |

#### Différentes manières de poster un formulaire

Il existe deux méthodes pour soumettre un formulaire de manière non intrusive. La méthode la plus répandue est d’utiliser le type « submit » de la balise « input ». La deuxième méthode, un peu moins connue, est d’utiliser le type « image » de la balise « input ». Ce type permet de soumettre un formulaire, au même titre que le « submit », mais il permet de définir l’apparence du bouton grâce à une image passée en attribut à cette balise, on peut aussi lui définir un attribut « alt » au cas où l’image n’est pas accessible.

# Table des Annexes

Annexe i - Commentaire jour à jour A

Annexe ii - Autre B

# Annexe

## Annexe i - Commentaire jour à jour

Passer par insertion -> légende -> intitulé Annexe

Annexe ii - Autre

# Table des matières

Remerciement 1

Sommaire 2

Table des illustrations 3

Abstract 5

Introduction 6

Présentation d’Atos 7

Sujet 8

Cycle de vie d’un projet 9

Le portail des Lorrains (PDL) 10

Présentation de Jahia 10

Développement 10

Syntaxe 11

Module 12

Template 12

Portlet 12

Les trois types de requêtes possibles sous Jahia 12

Le SQL 2 12

Le JQOM (Java Query Object Model) 12

Le XPATH 12

Travaux relatifs au Portail des Lorrains - version 1 13

FML 13

Conceptions — différentes solutions possibles : 14

Solution 1 14

Solution 2 15

Solution retenue 15

Problèmes rencontrés 16

Les remarques du client 16

ReadSpeaker 17

Version 2.0 du Portail des Lorrains 18

Déroulement du développement de la version 2 18

Outils utilisés 18

Problématique du découpage de certains éléments 19

En-tête du site 20

Logo 20

Accroche 20

Slider partenaire 20

Le framework ContentFlow 20

« Partenariat » avec le slider de la barre permanente 20

Respect du RGAA 21

Barre permanente 21

Problématique du respect du RGAA 21

Par rapport au JavaScript 21

« Développement parallèle » 21

« Développement procédural » 22

Par rapport au cahier des charges 22

Généralité sur l’« entrée par cible » et l’« espace privé » 22

Entrée par cible 23

Espace privé 23

Slider partenaire 23

Outil publicité 24

La cartographie des actions régionales 24

L’existant 24

Le souhait de la région 24

Le formulaire 24

La balise « label » 25

Les balises « fieldset » et « legend » 25

Modification de la requête SQL 25

La nouvelle carte 26

Recherche Synomia 26

Différentes manières de poster un formulaire 26

Table des Annexes 27

Annexe A

Annexe i - Commentaire jour à jour A

Annexe ii - Autre B

Table des matières A

1. Rédacteur du rapport : Nicolas Reitz [↑](#footnote-ref-1)
2. IE 8 et + [↑](#footnote-ref-2)
3. Firefox v10 et + [↑](#footnote-ref-3)
4. Format d’enregistrement par défaut de Photoshop [↑](#footnote-ref-4)
5. Red Green Blue Alpha ; la particule Alpha gère la transparence [↑](#footnote-ref-5)
6. HTC est l’extension d’un fichier qui permet la création d’un style CSS [↑](#footnote-ref-6)
7. Représente du JavaScript intrusif (bloquant) [↑](#footnote-ref-7)
8. Un élément formulaire fonctionne avec le JavaScript désactivé [↑](#footnote-ref-8)
9. Les weakreferences sont assimilables à des pointeurs [↑](#footnote-ref-9)
10. Internet Explorer 7 et plus, Firefox, Opera, Chrome, Safari [↑](#footnote-ref-10)