Atelier de programmation informatique pour les humanités numériques

Marc Douguet (chercheur post-doctoral)
Frédéric Glorieux (ingénieur de recherche)
Université Paris-Sorbonne, labex OBVIL

Public — chercheurs intéressés par la fouille des textes. Aucune notion de programmation n'est requise.

Objectif — apprendre les notions de base de la programmation informatique et de l'algorithmie, pour comprendre les principes de l'extraction et du traitement automatiques des données textuelles.

Méthode — les séances progresseront selon un cours classique d'informatique pour apprendre un langage (PHP), tous les exercices pratiques s'intéresseront aux textes littéraires. Les notions informatiques seront éclairées par une note historique et critique.

Logiciels — les participants sont invités à amener leurs propres ordinateurs et à y installer l'éditeur de texte Komodo Edit : http://www.activestate.com/komodo-ide/downloads/edit (logiciel gratuit).

[23/01/17]

1. Installation

Ce cours est une initiation au langage PHP, appliqué plus particulièrement aux questions de fouille de texte.

Contrairement à des langages comme HTML, XML, CSV, qui servent uniquement à stocker et à structurer des données, PHP permet d'aborder la programmation *impérative* : un script PHP est composé d'une série d'instruction que l'ordinateur exécutera dans l'ordre indiqué (afficher du texte, faire des opérations mathématiques, transformer un texte, créer, modifier ou supprimer un fichier, etc.).

D'autres langages reposent sur un principe comparable, et sont parfois plus puissants et efficaces. Les notions que nous étudierons (instructions, variables, conditions, boucles, etc.) sont communes à tous les langages impératifs. Pour une introduction à la programmation, PHP présente un double avantage :

– Il est simple à installer.

– C'est un langage qui est avant tout destiné à générer des pages web : l'acronyme PHP signifiait à l'origine *Personal Home Page*, et signifie désormais *PHP : Hypertext Preprocessor*. Une fois qu'on le maîtrise, on peut donc très facilement publier des applications et des visualisations en ligne. Nous commencerons à travailler "en local" sur nos propres ordinateurs, mais nous aborderons la question de la mise en ligne à la fin du cours.

Avant tout, il faut installer PHP.

1.1. Sous Mac

Il l'est déjà, vous pouvez passer au point 2.

1.2. Sous PC

1.2.1. Télécharger PHP

Déterminer si Windows est installé en 32 ou 64 bits : Explorateur de fichiers > Poste de travail > clic droit sur l'icône > Propriétés

– Système 64 bits : télécharger

http://windows.php.net/downloads/releases/archives/php-debug-pack-5.6.9-Win32-VC11-x64.zip

– Système 32 bits : télécharger

http://windows.php.net/downloads/releases/archives/php-debug-pack-5.6.9-Win32-VC11-x86.zip

Extraire le dossier php-5.6.9-Win32-VC11-x, le renommer php et le copier en racine. Le chemin de php.exe doit être C:\php\php.exe

1.2.2. Configurer le PATH

Menu Démarrer > Panneau de configuration > Système > Système > Avancé > Variables système > éditer "Path" > rajouter ;C:\php à la fin (avec un ; pour le séparer de ce qui précède)

2. Création d'un script PHP

Ouvrir Komodo, créer un nouveau fichier, et l'enregistrer sous le nom monscript.php, par exemple, dans un nouveau dossier, php, par exemple, que l'on crée sur le bureau. Sous Mac, on accède au formulaire d'enregistrement complet

(avec possibilité de créer un nouveau dossier) en cliquant sur la flèche à droite du nom du fichier.

On peut appeler le fichier et le dossier comme on veut, avec certaines règles communes à tous les fichiers informatiques. Exemple : comme / sépare les dossiers du chemin d'accès, on ne peut évidemment pas l'utiliser. Attention, on ne peut pas non plus utiliser d'espaces. La convention est d'utiliser uniquement les minuscules, les chiffres, et _

Deux choses sont nécessaires pour que le fichier soit reconnu comme un script PHP à exécuter :

– l'extension .php du fichier

– les balises PHP <?php (au tout début du fichier, sans espace entre les caractères) et ?> (à la toute fin). Tout le code PHP est placé entre ces balises : ne jamais rien écrire avant <?php ou après ?>.

3. Les instructions

3.1. Écrire le script

```
<?php
echo 'Hello world';
?>
```

echo : nom de l'instruction (ici, affiche le texte qui suit)

entre ' (guillemets simples) : le texte à afficher.

; : fin de l'instruction. Chaque instruction doit impérativement se terminer par ;

On découvre trois fonctionnalités très pratiques de l'éditeur :

– L'auto-suggestion (l'éditeur suggère le nom des instructions PHP qui existent)

– La coloration syntaxique (couleur différente pour le nom de l'instruction, les balises PHP, le texte.

– L'auto-fermeture des guillemets

Une étoile à droite du nom du fichier indique que des modifications ont été apportées mais ne sont pas encore enregistrées : enregistrer le fichier avant de l'exécuter.

Note sur les raccourcis clavier, plus rapides et plus directs que de cliquer sur une icône ou un menu : pour enregistrer, maintenir la touche ctrl (pour PC) ou cmd (pour Mac) enfoncée et appuyer sur s.

3.2. Exécuter le script

Pour exécuter le script, il faut passer par la console de commande.

– Sous Mac, ouvrir Terminal (dans le dossier Applications > Utilitaires).

– Sous PC, ouvrir l'invite de commande : menu Démarrer>Chercher>"Command" ou "cmd".

Note : épingler Komodo et la console dans la barre des applications ou créer un raccourci pour pouvoir les lancer facilement.

3.2.1. Vérifier que PHP est installé

Dans la console, écrire php -h et appuyer sur entrée (-h pour *help*). Si PHP est installé, une liste d'options s'affiche.

3.2.2. Se mettre dans le bon répertoire

Écrire cd Desktop/nom_du_dossier_ou_se_trouve_votre_scipt (par exemple cd Desktop/php) et appuyer sur entrée.

On se sent un peu paralysé face à la console, qui ne fonctionne pas comme les logiciels auxquels nous sommes habitués. Il est notamment impossible de cliquer (il faut utiliser les flèches "gauche" et "droite" pour se déplacer), et de modifier ce que l'on a précédemment écrit.

Tout ce que l'on fait en cliquant dans Finder ou Explorateur de documents (se déplacer dans l'arborescence des fichiers, ouvrir ou effacer un fichier, lancer une application), la console permet de le faire en écrivant

– une instruction (par exemple cd : change directory),

– des paramètres (le chemin du dossier où l'on veut se rendre, par exemple)

Instruction et paramètres doivent être séparés par des espaces

On exécute l'instruction en appuyant sur entrée. Une nouvelle "invite de commande" apparaît alors. Elle n'est pas modifiable, et indique

– sous PC, le chemin du dossier où je me trouve

– sous Mac, le nom du dossier où je me trouve et mon nom d'utilisateur

Le texte qui s'affiche au-dessus de l'invite de commande en cours n'est que l'historique des commandes que j'ai précédemment exécutées : il n'est donc pas modifiable. En cas d'erreur, il faut écrire et exécuter une nouvelle fois la commande en la corrigeant.

On prend conscience d'une des propriétés du langage informatique, qui est entièrement déterministe. L'ordinateur est incapable de comprendre une instruction erronée, si bénigne que soit l'erreur : je peux appeler comme je veux le dossier où j'ai placé mon script, mais il faut que la commande envoyée dans l'interpréteur de commandes corresponde au caractère près au nom du dossier, en respectant majuscules et minuscules – sinon l'ordinateur ne trouvera pas le dossier. Il en va de même à l'étape 1.2.2. (configuration du Path sous PC) : la moindre erreur dans l'écriture du chemin du dossier php empêchera d'exécuter du PHP. Ne pas oublier non plus la première partie de la commande (cd) ni l'espace qui la sépare de la suite.

3.2.3. Exécuter le script

La console permet également d'accomplir des actions que l'on ne peut pas accomplir par d'autres moyens, comme exécuter un script PHP.

Pour cela, lancer la commande php -f monscript.php (option -f pour *file* ; idem, ne pas oublier une partie de la commande, et respecter scrupuleusement le nom du fichier et les espaces de part et d'autre de -f)

PHP exécute les instructions contenues dans le script. Pour les instructions d'affichage (comme ici), le résultat est affiché dans la fenêtre de la console.

Cela paraît bien compliqué, mais afficher simplement un fichier texte et exécuter un script qui affiche un texte sont deux opérations très différentes. echo est loin d'être la seule instruction possible : il en existe bien d'autres, qui nous permettront de traiter et de modifier les données avant de les afficher.

3.3. Un script avec deux instructions

On revient à Komodo, et on ajoute une seconde instruction.

```
<?php
```

```
echo 'Hello world!';
echo 'Hello, you!';
?>
```

PHP va exécuter successivement les deux instructions, l'une après l'autre.

Enregistrer monscript.php dans Komodo (l'étoile signalant des modifications non enregistrées disparaît), et revenir à la console pour l'exécuter. On peut retrouver les commandes précédemment exécutées en navigant avec les flèches "haut" et "bas" : il ne reste qu'à valider en appuyant sur entrée.

Note sur les raccourcis clavier

Nous allons, dans ce cours, alterner en permanence entre Komodo (pour modifier notre script) et la console (pour l'exécuter). Ne jamais fermer ou réduire les fenêtres de ces deux applications. Pour passer rapidement de l'une à l'autre, maintenir la touche alt (sur PC) ou cmd (sur Mac) et appuyer sur la touche tabulation (⇥) et sélectionner l'application voulue en appuyant autant de fois que nécessaire sur la touche ⇥ (tout en maintenant alt ou cmd enfoncée).

Après avoir modifié le script, on exécutera donc toujours les quatre mêmes actions :

- ctrl ou cmd s (enregistrer)
- alt ou cmd ⇥ (revenir à la console)
- flèche "haut" (retrouver la commande précédente, `php -f monscript.php`)
- entrée

3.4. Les sauts de lignes

echo affiche uniquement les caractères entre guillemets. Les sauts de ligne du script ne sont pas à l'intérieur de l'instruction, donc ils ne sont pas affichés.

– On peut en mettre autant qu'on veut

– On peut en rajouter (après echo, avant ;)

Mais convention et bon sens : une instruction entière et une seule par ligne

Si je veux afficher un saut de ligne ? dans Terminal, la touche entrée est réservée pour l'exécution, donc les sauts de ligne sont codés \n. echo 'Hello world ! \n'; Il faut également en afficher un à la fin du texte pour que la nouvelle invite de commande ne soit pas sur la même ligne que la dernière ligne du texte affiché.

3.5. Deux langages coexistent dans le fichier, deux types de mots

3.5.1. La langue naturelle du texte affiché

Un lexique et un syntaxe, compréhensible par les humains. Des erreurs sont possibles sans faire échouer la communication. On peut créer des mots tout en se faisant comprendre.

3.5.2. Le langage PHP

Totalement déterministe. Marge de liberté très faible dans la syntaxe et le lexique : si j'omets un seul caractère, tout peut planter. Lexique restreint.

Testons la souplesse de PHP

3.5.2.1. Espaces

Il y a une certaine tolérance dans l'emploi des espaces. On pourrait écrire echo'Hello world!' ; au lieu de echo 'Hello world';

C'est la convention, la lisibilité, l'économie qui nous guident.

Les balises PHP s'écrivent <?php et ?>, sans espace entre les caractères.

3.5.2.2. "Mot" php incorrect

– eco 'Hello world !'; L'instruction n'existe pas. On découvre une autre fonctionnalité de l'éditeur de texte : les erreurs sont soulignées (première aide au déboggage). Les messages d'erreur renvoyés par PHP sont une deuxième aide : indiquent où se trouve l'erreur, et sa nature

3.5.2.3. "Mot" php oublié

– les ; Si j'oublie le dernier, ça passe. Mais si j'oublie les précédents, PHP n'a aucun moyen de distinguer les instructions successives

3.5.3. Question : comment l'ordinateur distingue-t-il ces deux langages ?

Le caractère réservé ' délimite les segments qui contiennent des données textuelles de ceux qu'occupe le langage php.

C'est une "chaîne de caractère" (*string*, en anglais)

On comprend donc qu'il soit impossible d'employer ' à l'intérieur d'une chaîne de caractère

echo 'Bonjour l'univers !'; > erreur

mais echo 'Bonjour l'univers !'; (alt 4 sous Mac ou copier de Word : Word remplace automatiquement ' en ', ce que ne fait pas l'éditeur de texte ; alt 0146 sous PC)

Sous Windows, les problèmes d'encodage sont réglés en exécutant la commande chcp 65001

Il faut donc "échapper" le caractère réservé en le faisant précéder de \ pour indiquer qu'il fait pleinement partie de la chaîne de caractère et que PHP ne doit pas le comprendre comme un délimiteur : echo 'Bonjour l\'univers'; (shift alt / sous Mac, ctrl alt 8 sous PC)

3.6. Les commentaires

PHP saute les lignes qui commencent par //. Elles restent visibles pour nous dans le fichier, mais pour PHP, c'est comme si elles n'existaient pas.

Cela permet

– De faire des tests ou de garder une trace des instructions précédentes

– D'introduire des commentaires dans son code, qui permettent d'en comprendre rapidement la logique (utile quand on revient sur un code écrit il y a longtemps, ou par quelqu'un d'autre)

Pour le cours, permettra de prendre des notes et de garder une trace des états antérieures du script (raccourci = ctrl / ou cmd /)

4. 30 janvier

une fonction

en entrée, en sortie

les données sont stockées dans différents types de variables : texte, entier, booléen, tableau (array)

nom de la variable : $

contenu assigné à cette variable grâce à l'opérateur = : c'est aussi une instruction

$bonjour = '';

print $bonjour;

nommage : fonctionne comme un "mot" php : pas d'espace, et ne peut contenir de caractères spéciaux utilisés pour d'autres usages par php : ';= (entre autres)

comme les noms de fichiers, donc (mais pas de point)

conditions if/else/elseif et opérateurs de comparaison

5. **6 février**

   tableaux numériques et associatifs (unidimensionnels)
   boucles while et foreach

6. **20 février**

   formulaire html
   fonctions natives sur les chaînes de caractères
   lire et écrire un fichier

7. **27 février**
8. **6 mars**
9. **13 mars**
10. **20 mars**
11. **27 mars**
12. **24 avril**