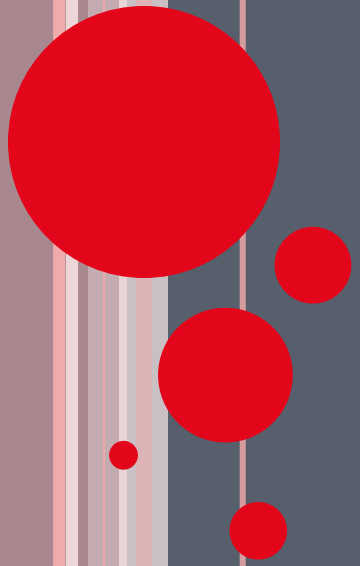
académie Aix-Marseille
GRETA PROVENCE
FORMEZ-VOUS À DEMAIN

PHP 5-7

Découvrir la programmation orienté objet et la conception MVC

Connaître les bases du Framework Symfony

Maîtriser le Framework de persistance Doctrine



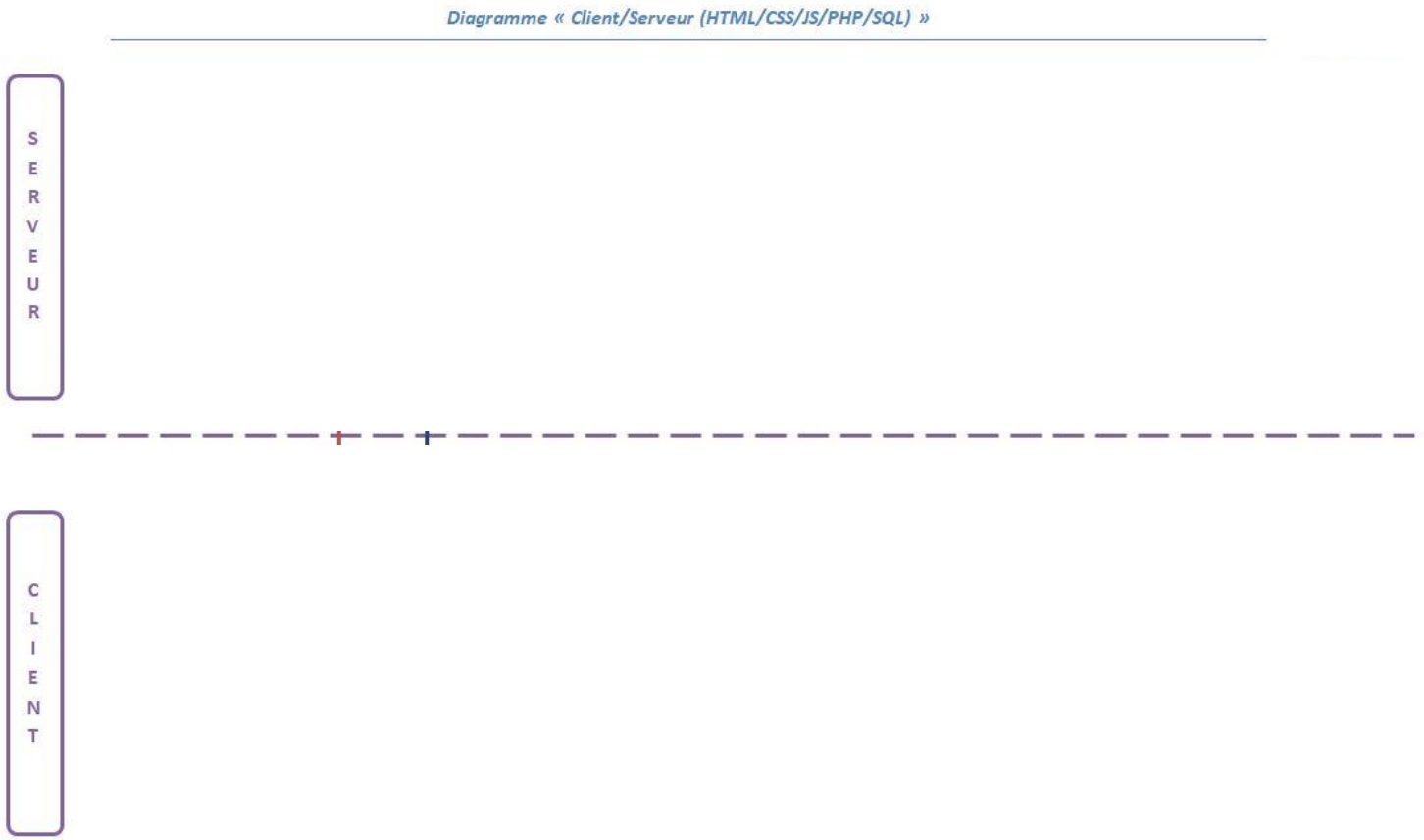
Rappels

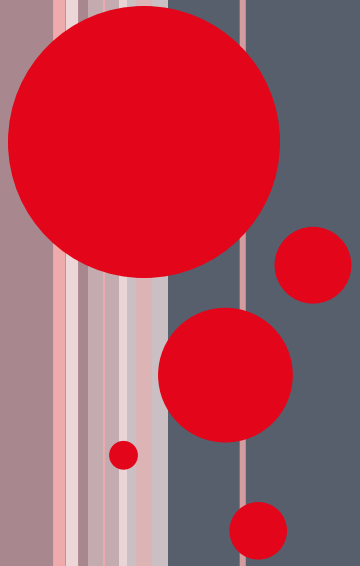
HTML5 / CSS3 / JS / jQuery / Bootstrap

PHP / SQL

Cookies et sessions

RAPPELS CLIENT-SERVEUR





PHP

Historique et rappels

De PHP4 à PHP 7.4

- PHP 4 (2000)
  - Le constructeur a le nom de la classe
  - Pas de vraie notion « objet »
  - Passages des variables par valeur

- PHP 5 (2004) - PHP 5.6 (2014)
  - Le constructeur s'appelle __construct ( )
  - Notion « objet » (visibilité des attributs, classes abstraites et finales)
  - Passages des objets par référence
  - Héritage, interfaces et traits (PHP 5.4)
  - Exceptions

De PHP4 à PHP 7.4

- PHP 7.0 (2015)
  - Amélioration des performances (2x)
  - Type de retour sur les fonctions
  - Spaceship operator (<=>)
  - Null Coalesce Operator (??)
  - Classes anonymes (function () { … })
- PHP 7.1 (2016)
  - Type de retour « void » et « null possible » (hello() : ?int)
  - Extraction de tableaux avec des crochets ([$a, $b] = $aArr)
  - Visibilité des constantes de classes

De PHP4 à PHP 7.4

- PHP 7.2 (2017)
  - Nouveau type « object »
- PHP 7.3 (2018)
  - Syntaxes heredoc/nowdoc améliorées
  - Virgule autorisée en fin de fonction
  - array_key_first( ) / array_key_last( )
- PHP 7.4 (2020)
  - Propriétés typées (public string $name)
  - Fonctions flèches (array_map(fn($n) => $n * $factor, [...]))
  - Opérateur d'assignement de fusion Null (??=)
  - Déballage dans les tableaux (['AA', 'BB', ...$aParts, 'CC'])

Les bases du langage PHP (1/2)

- Langage de programmation utilisé principalement pour produire des pages web dynamiques

- Concepts communs
  - Variables, types simples et complexes
  - Opérateurs (+ - / * % == === != !== ++ -- && || !)
  - Structures de contrôles
    - Les conditions (if/else ; switch/break)
    - Les boucles (while ; do..while ; for ; foreach)
  - Les fonctions
    - Paramètres et valeur de retour
    - Notion de « scope » (portée des variables)

Les bases du langage PHP (2/2)

- Implémentations propres à PHP
  - Tableaux
    - Parcours de tableau : foreach ($aArray as $iKey => $sVal)
    - array_key_exists (..) ; in_array (..) ; array_search (..)
  - Affichage et débogage
    - echo 'Bonjour Fabien !';
    - print_r ($variable)
  - Concaténation de variable
    - $sVar = $sStr1 . ' lorem epsum ' . $sStr2;
    - $sVar .= $sStr3;
  - Inclusion de fichiers
  - Le type NULL

Exercice 1

- Créer un programme exécutable en ligne de commande
  - Le programme doit demander le prénom de l'utilisateur et lui dire « Bonjour {Prénom} ! »

```
F2000@F2000-PC MINGW64 /d/Formation/PHP-PE
$ php exo1.php
== Début du programme ==
Quel est votre prénom ? Fabien
Bonjour Fabien !
== Fin du programme ==
```

  - readline (..) demande une saisie à l'utilisateur

PHP et la POO

Classes, objets, méthodes et propriétés
- Visibilité des attributs
- Le constructeur
- L'héritage , les interfaces et les traits

Gestion des exceptions

Les espaces de nommage

Les classes

- Objet « complexe » qui permet de représenter de entités ayant des propriétés et des comportements
  - Les propriétés sont représentées par des attributs
  - Les comportements sont représentés par des fonctions

- Notion de visibilité
  - public (*par défaut* ; accessible depuis l'extérieur)
  - protected (accessible au travers de l'héritage)
  - private (non accessible hormis au sein de la classe)

- Classe = référentiel de construction
- Instance = exemplaire unique de la classe (entité)

Les classes

Exercice 2

- Créer deux classes « Warrior » et « Wizard »
  - Chaque instance de Warrior doit posséder
    - Un nom, des points de vie, des points de force
  - Chaque instance de Wizard doit posséder
    - Un nom, des points de vie, des points de force, des points de magie

```
F2000@F2000-PC MINGW64 /d/Formation/PHP-PE
$ php exo2.php
== Début du programme ==
War - [H: 100] - [S: 10]
Wiz - [H: 100] - [S: 10]- [M: 10]
== Fin du programme ==
```

L'HÉRITAGE

- Permet de simplifier et d'optimiser la conception des classes
  - class Guerrier extends Personnage
- Une classe peut hériter d'une autre classe et ainsi récupérer ses propriétés et ses comportements publiques ou protégés
  - On parle alors de classe fille et de classe parente
- Classes abstraites et finales
  - abstract class
  - final class

L'HÉRITAGE

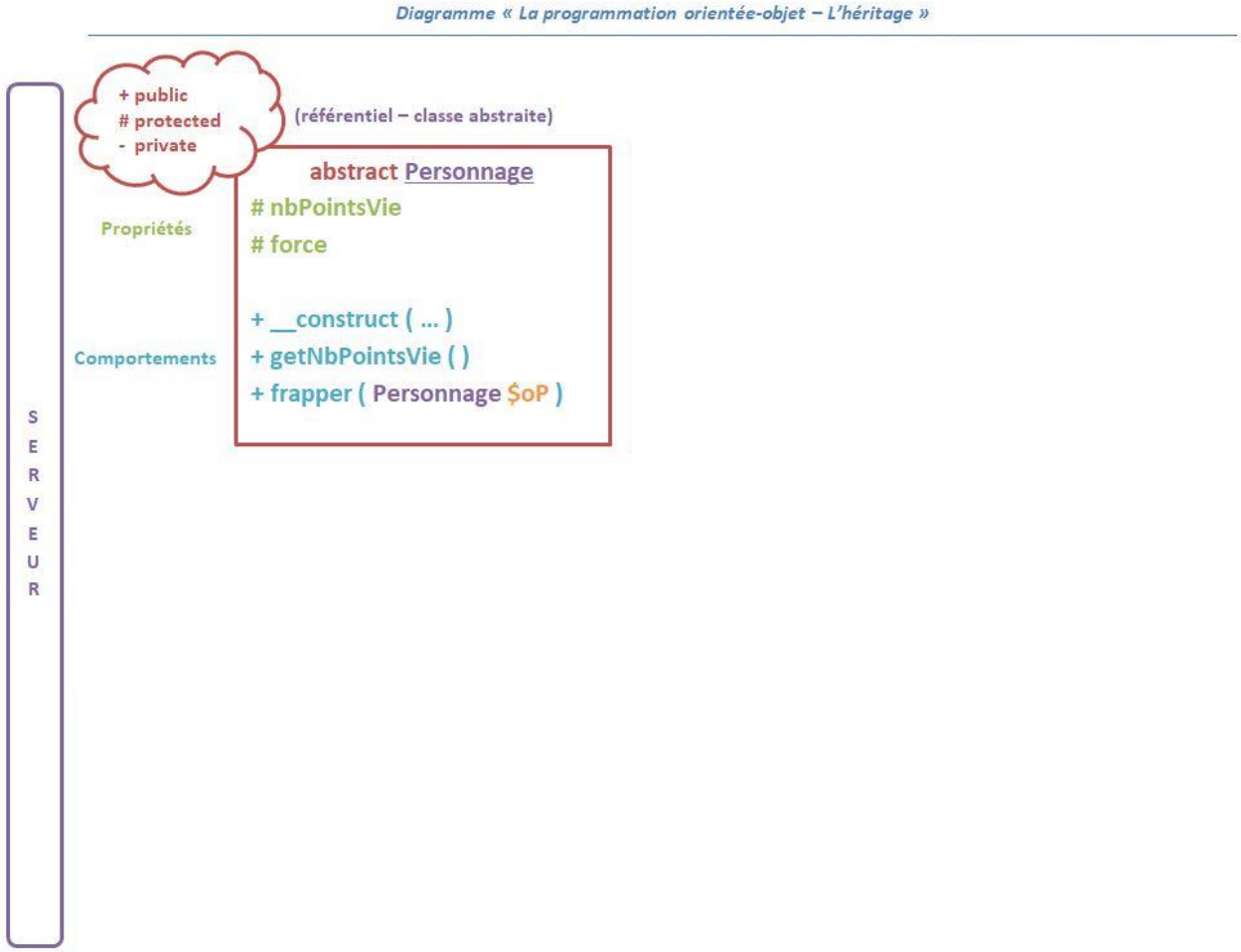


Diagramme « La programmation orientée-objet – L'héritage »
SERVEUR
+ public
# protected
- private
(référentiel – classe abstraite)
abstract Personnage
Propriétés
nbPointsVie
force
Comportements
+ __construct (...)
+ getNbPointsVie ( )
+ frapper (Personnage $oP)

Exercice 3

- Créer la classe « Character »
  - Reprendre les points communs des classes « Warrior » et « Wizard »

```
F2000@F2000-PC MINGW64 /d/Formation/PHP-PE
$ php exo3.php
== Début du programme ==
War - [H: 100] - [S: 50]
Wiz - [H: 100] - [S: 10]- [M: 10]
== Fin du programme ==
```