**JEE**

request.getParameter(‘’x’’) 🡪 permet de récupérer la variable x de la requête.

PrinterWriter p = response.getWriter() ; p.println(“hello world”); 🡪 Ecrire dans la reponce HTTP.

request.getRequestDispatcher(“/index”).forward(req,res);🡪 permet de déléguer le traitements á une autre servlet locale, sans changement au niveau de l’URL coté client, c’est comme si la première servlet qui a fait le traitement.

request.setAttribute(‘’k’’,5) ; request.getAttribute(‘’k’’) 🡪 permet de passer et de récupérer des variables d’une servlet á une autre, lors de l’utilisation d’un dispatcher.

responce.sendRedirect(‘’http://www.google.com’’) ; 🡪 permet de faire une redirection, il s’agit d’une nouvelle requête, du coup les variables de la 1ere requête ne sont plus valable.

Session :

HttpSession s = request.getSession() 🡪 permet d’instancier une variable de session.

s.setAttribute(‘’k’’,3) 🡪 permet d’attribut une valeur á une variable de sessions.

s.removeAttribute(‘’k’’) 🡪 pemet de supprimer la variable de session k.

int k = (int)sess.getAttribute(‘’’k’’) 🡪 permet de récupérer la valeur de la variable de session k.

Cookies :

Cookie c = new Cookie(‘’k’’,5) 🡪 créer est attribué une valeur á un cookie.

c.setMaxAge(3600) 🡪 permet de spécifier la durée de vie d’un cookie.

response.addCookie(c) 🡪 pour demander de stocker le cookie sur le disque dur du client.

c.getName() 🡪 permet de récupérer le nom du cookie.

c.getValue() 🡪 permet de récupérer son contenu.

request.getCookies() 🡪 retourne un tableau des cookies.

@WebServlet(‘’/index’’) 🡪 utiliser pour mapper une servlet par un URL, du coup ici la servlet serait solliciter á chaque fois que la requête rediriger vers le lient ‘’/index’’.

<%@page import=’’java.util.Date’’ %> 🡪 permet d’importer un package au sein d’un page JSP.

<%@include file = ‘’header.jsp’’ %> 🡪 permet d’inclure un autre fichier jsp dans le notre.

<%@taglib uri = ‘’http://java.sun.com/jsp/jstl/core’’ prefix = ‘’c’’ %> 🡪 inclure jstl.

JDBC

Tout d’abord il faut mettre le driver dans le dossier /WEB-INF/lib.

Class.forName(‘’com.mysql.jdbc.Driver’’) 🡪 permet de charger la classe driver au niveau de la mémoire.

connetion = DriverManager.getConnextion(‘’jdbc :mysql://localhost :3306/EtudiantBase’’, ‘’root’’, ‘’’’) 🡪créer l’objet connexion,

preparedStatement = connection.prepareStatement(‘’ query ?,?’’) 🡪 préparer une requête.

preparedStatement.setString(1,’’toto’’) 🡪 permet de remplacer la première ? par toto.

preparedStatement.setInt(2,99) 🡪 remplacer deuxième ? par 99.

preparedStatement.executeUpdate() 🡪 pour exécuter une requête de type DML.

resultSet = preparedStatment.executeQuery() 🡪 exécuter une requête de sélection, et que resultSet va contenir le résultat de la requête.

resultSet.next() 🡪 permet de parcourir á chaque appel le résultat de la requête ligne par ligne.

resultSet.getString(‘’columnName’’), getInt …🡪 permet de récupérer la valeur qui correspond á la colonne préciser en paramètres.

JSTL

< c :out value = ‘’${k}’’ /> 🡪 permet d’afficher la valeur de la variable k.

< c :forEach item = ‘’patients’’ var = ‘’p’’> ${p.name} </ c:forEach > 🡪 permet de boucler sur une liste.

< c :if test = ‘’${ expression }’’ var = ‘’x’’ > ${x} </ c:if> 🡪 La valeur x sera afficher, s’elle satisfait la condition ‘’expression’’.

@WebFilter(‘’/index’’) 🡪utiliser au niveau des filter pour intercepter les requêtes dirigeant vers l’URL ‘’/index’’

response.setContentType(‘’text/html’’) 🡪 permet de dire au navigateur que le contenu de la réponse et de type html text/html.

request.getServletPath() 🡪 utiliser pour séparer les traitements au niveau d’une servlet, il permet de retourner le chemin á partir de la requête.

request.getMethode() 🡪 retourne la méthode HTTP utiliser au niveau de la requête (POST, GET).

response.sendError(Response.SC\_NOT\_FOUND) 🡪 utiliser pour dire au client le ressource que vous cherchez est introuvable.

**PHP**

global $V1 ; $GLOBALS[‘$V’] ; 🡪 permet d’utiliser le contenu de la variable globale $V1 au sein d’une fonction.

isSet($v) 🡪 vérifier si la variable est déjà créer et qu’il a un contenu, sinon il renvoie 0.

define(P, 3.14) , const P = 3.14 🡪 permet de déclarer une constante.

include(‘fichierEmplacement’) 🡪 permet de charger le code de fichier passer en paramètres.

require\_once(‘fichierEmplacement’) 🡪 comme include, sauf si le fichier n’existe pas le script arrête.

file\_exists(‘fichierEmplacement’) 🡪 renvoie true si le fichier existe.

die(‘message’) 🡪 permet d’arrêter l’exécution en affichant un message.

exit() 🡪 permet d’arrêter l’exécution de script.

getType($v1) 🡪 retourne le type de la variable $v1 ;

setType($v1,$type) 🡪 permet de changer le type de la variable $v1.

echo ‘’texte’’, print(‘’texte’’) 🡪 permet d’afficher du texte combiné avec des variable.

printf(‘’text’’) 🡪 a le même comportement de printf du langage C.

is\_string($v1), is\_numeric, is\_double … 🡪 retourne true si la variable s’agit de type indiqué.

empty($v) 🡪 test le contenu de $v, renvoie true s’il s’agit d’une chaine vide ou 0.

unSet($v) 🡪 permet de libérer $v de la mémoire.

getDate() 🡪 retourne un tableau associatif qui contient les détails de la date actuelle.

date(‘D d/m/y H : i :s’) 🡪 Wed 10/01/2021 20:13:39.

checkDate($mois, $jour, $annee) 🡪 renvoit true s’il s’agit d’une date valide.

rand($min, $max) 🡪 retourne une valeur aléatoire entre le min et le max.

Les chaines des caractères :

strlen($v) 🡪 retourne la taille d’une chaine.

strToUpper($v) 🡪 transforme la chaine en majuscule.

strToLower($v) 🡪 transforme la chaine en minuscule.

trim($v) 🡪 permet de supprimer des espaces de début et de fin de la chaine.

substr($s,$d,$f) 🡪 permet de retourner la sous chaine qui est entre d et f.

strcmp($s1,$s2) 🡪 permet de comparer alpha numériquement le contenu de s1 avec s2.

addSlaches($s) 🡪 si la chaine s contient des caractères spéciales, il va les déspécialisé.

Les tableaux :

$tab = array($e1, $e2, $e3) 🡪 permet de retourner un tableau contenant les éléments e1, e2 ,e3.

$tab = [1, 2, 3, 5] 🡪 création d’un tableau.

$tab[] = 6 🡪 ajouter 6 á la fin du tableau.

count($tab) 🡪 retourne la taille de tableau ;

in\_array($v, $tab) 🡪 vérifier si la valeur de $v existe dans le tableau.

list($v1, $v2, $v4 ) = tab 🡪 permet d’enregister les éléments de tab dans $v1, $v2 ..

sort($tab) 🡪 trie les éléments du tableau.

rsort($tab) 🡪 trie inverse des éléments du tableau.

implode($s, $tab) 🡪 joindre les éléments du tableau dans une chaine de caractéres.

array\_merge($t1, $t2, … ) 🡪 concatener les tableau t1, t2, … en un seul tableau.

array\_rand($tab) 🡪 renvoie un élément aléatoire du tableau.

array\_count\_value($tab) 🡪 retourne un tableau associatif qui contient comme des clés les éléments du tableau, et comme valeur leur occurrences au niveau du tableau.

array\_keys($tab) 🡪 retourne un tableau contenant les clés du tableau associatif.

array\_values($tab) 🡪 retourne un tableau contenant les valeurs du tableau associatif.

Les fichiers :

$fic = fopen(‘emplacement’,$mode) 🡪 permet d’ouvrir un fichier.

fclose($fid) 🡪 fermer le fichier $fic.

fgetc($fic) 🡪 lit caractère par caractère de fichier $fic.

fgets($fic) 🡪 lit ligne par ligne de á partir de $fic.

fputs($fic, $s) 🡪 permet d’écrire la chaine $s dans le fichier $fic.

feof($fic) 🡪 teste la fin du fichier $fic.

fileSize($fic) 🡪 retourne