Stateless vs stateful protocol

- **Stateless** versus **stateful** protocol
  - Stateless:
    - Independant request – no memory
    - More info per request (ex.: cookies)
  - Stateful:
    - Keep track of the dialog status
    - Possibility for pause/resume
- FTP, HTTP: stateless
- Solutions:
    - Client side data= **cookies** (as seen with JS)
    - Server side data = **sessions**

cookies

- Small text files (size <4KB)
- When? Before all the markups (<html>,etc)
- Create? (optional parameters)

```
setcookie(name, value, duration, domain)
ex.: setcookie("myname","gilles",time()+3600,".irit.fr")
```

- Delete ? Set with a past date

```
setcookie(name) //or past date
ex.: setcookie("myname")
```

- Read ?

```
$_COOKIE[name]
ex.: echo $_COOKIE["myname"]
echo $_COOKIE //all the cookies
if (isset($_COOKIE["myname"])) echo "hello ".$_COOKIE["myname"])
```

Basic example!

- Create then delete

```
$date=time()+3600;
 setcookie("myname","gilles",$date);
 echo "we have set up a cookie!<br/>";
 if (isset($_COOKIE["myname"]))
  {echo "Welcome " . $_COOKIE["myname"]."!<br
/>";}
      else {echo "Welcome!<br/>";};
   //setcookie("myname","",time()-3600);
   //echo "cookie deleted !<br/>";
if (isset($_COOKIE["myname"])){
                  echo "still alive...<br/>";}
       else {echo "deleted....<br/>";}
```

Login scripts with cookies

```
//if OK for login
$date = time() + 3600; //1h session
setcookie("yourname", $_POST['username'], $date);
$header="location:member.php?name=".
$_POST['username'];//redirect to member
header($header); }
//member
if (isset($_COOKIE['yourname'])) {
echo "Welcome on board "…
   else {//go to login.php}
//logout
$date = time() - 3600; //past date
setcookie("yourname", "", $date);
header("Location: login.php");
```

Pros/cons

- **pros**
  - Client/server communication
  - Human-like dialog
  - UX (automatic login for instance)
- **cons**
  - Write on the client drive
  - privacy
  - Security
  - Can be forbidden (not reliable)
- **Conclusion: we need something else;-)**

Sessions...

- **Start ?** `session_start()`
  - **UID stored server side + cookie client side**
  - **Where**: see `session.save_path` (php.ini)
  - **Limited lifetime**
  - **At the beginning of the page**
- **Stop ?** `session_destroy()`
- **Usage ?**
  - **Global Variable (array)** : `$_SESSION`

```
$_SESSION[“myname”] = $_POST[“”]
unset($_SESSION[“myname”])//delete
if isset(...) to check if set
```

Basic example

- **No cookie anymore (except PHPSESSID if default)**
- **Login process**
  - **Idea**:
    - Ask for login/pass
    - Check db
    - if yes then `session_start()` and member page
    - if not then register.php (no change)
    - Logout : `session_destroy()`
    - Cookies to be destroyed as well if any
- **Just do it ;-)**

Login script now

```
//if ok
session_start(); //can set up a cookie PHPSESSID
$_SESSION['name']=$_POST['username'];
$header="location:member.php
header($header); //as usual
}
else
{ as usual.....................
```

Member script now

```
session_start();//to check the session variables
if any – start or resume a session
if (isset($_SESSION['name'])) {
echo "Welcome on board ". $_GET['nom']." <br/>";
echo "Admin Area<br>";
echo "Your Content<br>";
echo "<a href=logout.php>Logout</a>";}
else { //session_start() has set a cookie to be
destroyed – default behavior
   setcookie("PHPSESSID","",time()-3600,"/");
   session_destroy();
   header("Location: login.php"); }
```

Logout script now

```
session_start();
if (isset($_SESSION['name']) ) {
    unset($_SESSION['name']);
};
setcookie("PHPSESSID","",time()-3600,"/");//in
any case we destroy the cookie
session_destroy(); //created with session_start
header("Location: login.php");
```

Conclusion

- Session : the main tool to manage client/server communication
- Implement a kind of stateful-like protocol
- Properties/methods of the object
- What is missing ?
  - Ask for just a part of a page
  - Decrease server /network load
- New tool needed **AJAX**