



# **Gestione delle sessioni**

## **Corso di Laboratorio di Informatica**

Prof. Dei Rossi, Leonardo Essam

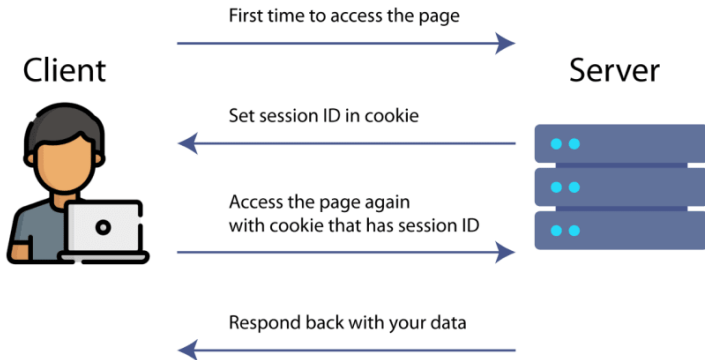
## Che cos'è una sessione? (1)

**Il protocollo HTTP, che sovrintende tutte le connessioni web, è *stateless*: ovvero non è in grado di tenere traccia di eventuali scelte o particolari dati trasmessi tra client e server.**

Per questa ragione in PHP è stato implementato un meccanismo di **gestione delle sessioni**, laddove per "sessione" si intendono tutte le interazioni avvenute tra browser e server fino alla chiusura del browser stesso.

## Che cos'è una sessione? (2)

In questo modo si potrà tenere traccia di eventuali precedenti scelte fatte dall'utente come, ad esempio, i prodotti scelti in un carrello oppure l'essersi autenticato in un'area riservata.



# Le sessioni in PHP (1)

## Definizione 1: Variabile `$_SESSION`

`$_SESSION` è un array associativo globale che contiene al suo interno i dati della sessione corrente salvati nel formato chiave-valore.

Le sessioni in PHP sono quindi uno strumento fondamentale per salvare dei dati *per-utente* e quindi poter personalizzare l'esperienza di navigazione del singolo utente.

## Le sessioni in PHP (2)

Per inizializzare una sessione in PHP (sia essa nuova o pre-esistente), vale il comando:

```
<?php  
session_start();  
?>
```

La funzione `session_start()` va a verificare se nel browser è presente un cookie nel quale vi è contenuto l'ID della sessione, seguono quindi due scenari.

## Le sessioni in PHP (3)

### Scenario 1

La funzione `session_id()` trova il cookie richiesto e carica la sessione corrispondente. Quindi all'interno dell'array `$_SESSION` verranno pre-caricate le variabili necessarie e l'utente potrà riprendere da dove aveva interrotto.

---

### Scenario 2

La funzione `session_id()` non trova il cookie richiesto e quindi ne inizializza uno nuovo con un nuovo ID di sessione. Segue che l'array `$_SESSION` sarà vuoto.

# Il cookie di sessione (1)

## Definizione 2: Cookie

Un cookie è una **coppia chiave-valore** (con metadati associati) definita nel protocollo HTTP, utilizzata per **mantenere lo stato applicativo** in un sistema di comunicazione stateless.



## Il cookie di sessione (2)

**Per impostazione predefinita, il nome del cookie di sessione è PHPSESSID.**

È possibile però modificare a piacimento il nome del cookie nel seguente modo:

```
<?php  
session_name("pino");  
session_start();  
?>
```

Allo stesso modo si può leggere il nome del cookie di sessione con la stessa funzione:

```
<?php echo(session_name()); ?>
```



## Il cookie di sessione (3)

Il poter modificare il nome del cookie di sessione è utile in quei scenari dove sullo stesso dominio convivono due o più applicazioni. Per evitare che le sessioni vadano in collisione si può quindi dare un nome a ogni sessione.

Nome	Valore
PHPSESSID	33cb78dd7e058e0c4ccc953cd439ebc3
pino	ad7ea01b8c3f5f3c38d6f8c83d2cd233

**La scadenza di un cookie di sessione corrisponde alla chiusura della finestra del browser.**

## Lavorare con le sessioni (1)

La lettura e la scrittura di una variabile di sessione seguono le stesse regole degli array chiave-valore.

### Esempio 1: Lettura di una variabile di sessione

```
<?php  
echo("Ciao, " . $_SESSION["nome"] . "!");  
?>
```

---

**Output:** Ciao, Leonardo Essam!

## Lavorare con le sessioni (2)

### Esempio 2: Scrittura di una variabile di sessione

```
<?php
echo("Ciao, " . $_SESSION["nome"] . "!");

$_SESSION["nome"] = "Laura";

echo("Ciao, " . $_SESSION["nome"] . "!");
?>
```

---

#### Output:

```
Ciao, Leonardo Essam!
Ciao, Laura!
```

## Lavorare con le sessioni (3)

Per eliminare una variabile di sessione, ovvero eliminare una coppia chiave-valore, vale la funzione `unset ( [ . . . ] )`.

### Esempio 3: Eliminazione di una variabile di sessione

```
<?php  
$_SESSION["nome"] = "Leonardo Essam";  
unset($_SESSION["nome"]);  
?>
```

## Lavorare con le sessioni (4)

**Si presti particolare attenzione al verificare l'effettiva esistenza delle variabili di sessione.**

Se dopo aver invocato la funzione `unset ( [ . . . ] )` come visto nell'**Esempio 3** si prova ad accedere alla variabile di sessione "nome" verrà restituito un *warning*:

**Warning:** Undefined array key "nome" in `/var/www/html/cookie.php` on line 11  
Ciao, !

## Lavorare con le sessioni (5)

È possibile però verificare se una data variabile di sessione esiste o meno tramite la funzione `key_exists([...])`.

### Esempio 4: Esistenza di una variabile

```
<?php
$_SESSION["nome"] = "Leonardo Essam";
var_dump(key_exists("nome", $_SESSION));
var_dump(key_exists("cognome", $_SESSION));
?>
```

---

#### Output:

```
bool(true)
bool(false)
```

## Eliminare una sessione (1)

Una volta terminato il lavoro (ad esempio quando si esegue il log-out), è possibile eliminare tutti i contenuti dalla sessione memorizzati nel server, rendendo di fatto inutile il cookie salvato nel computer.

### Esempio 5: Eliminazione della sessione

```
<?php
// [...]

session_unset();
session_destroy();
?>
```

## Eliminare una sessione (2)

La combinazione delle funzioni `session_unset()` e `session_destroy()` svolge due operazioni:

1. Elimina tutte le variabili di sessione;
2. Elimina la sessione in sé.

**Domanda:** perché non facciamo direttamente `session_destroy()`?



