

# PHP Sessions

- A session is a way to store information (in variables) to be used across multiple pages.
- Unlike a cookie, the information is not stored on the users computer, but sessions do internally rely on cookies in order to identify who is connecting

# PHP Sessions Demo

- How to use sessions with demos.

DEMO 02

```
mirror_mod = modifier_ob.  
# mirror object to mirror  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
selection at the end -add  
mirror_mod.select= 1  
modifier_ob.select=1  
context.scene.objects.active =  
("Selected" + str(modifier))  
mirror_ob.select = 0  
bpy.context.selected_objects =  
data.objects[one.name].select  
  
print("please select exactly one object")  
-----  
-- OPERATOR CLASSES -----
```

```
types.Operator):  
    X mirror to the selected object.mirror_mirror_x"  
    "mirror X"  
  
context):  
    context.active_object is not None
```

# PHP Demo 02a Session

## Start a PHP Session

- A session is started with the `session_start()` function.
- Session variables are set with the PHP global variable: `$_SESSION`.
- **Note:** The `session_start()` function must be the very first thing in your document. Before any HTML tags.

# PHP Demo 02a Session

- Now, let's create a new page called "demo\_session1.php". In this page, we start a new PHP session and set some session variables:

```
<?php
    // Start the session
    session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
    // Set session variables
    $_SESSION["favcolor"] = "green";
    $_SESSION["favanimal"] = "cat";
    echo "Session variables are set.";
?>
</body>
</html>
```

# PHP Demo 02a Session

- **Get PHP Session Variable Values**

- Next, we create another page called "demo\_session2.php". From this page, we will access the session information we set on the first page ("demo\_session1.php").
- Notice that session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page (`session_start()`).
- Also notice that all session variable values are stored in the global `$_SESSION` variable:
  -

# PHP Demo 02a Session

## • Get PHP Session Variable Values

```
<?php
    session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
    // Echo session variables that were set on previous page
    echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
    echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>
</body>
</html>
```

# PHP Demo 02a Session

## Modify a PHP Session Variable

- To change a session variable, just overwrite it:

```
<?php  
    // to change a session variable, just overwrite it  
    $_SESSION["favcolor"] = "yellow";  
?>
```

# PHP Demo 02a Session

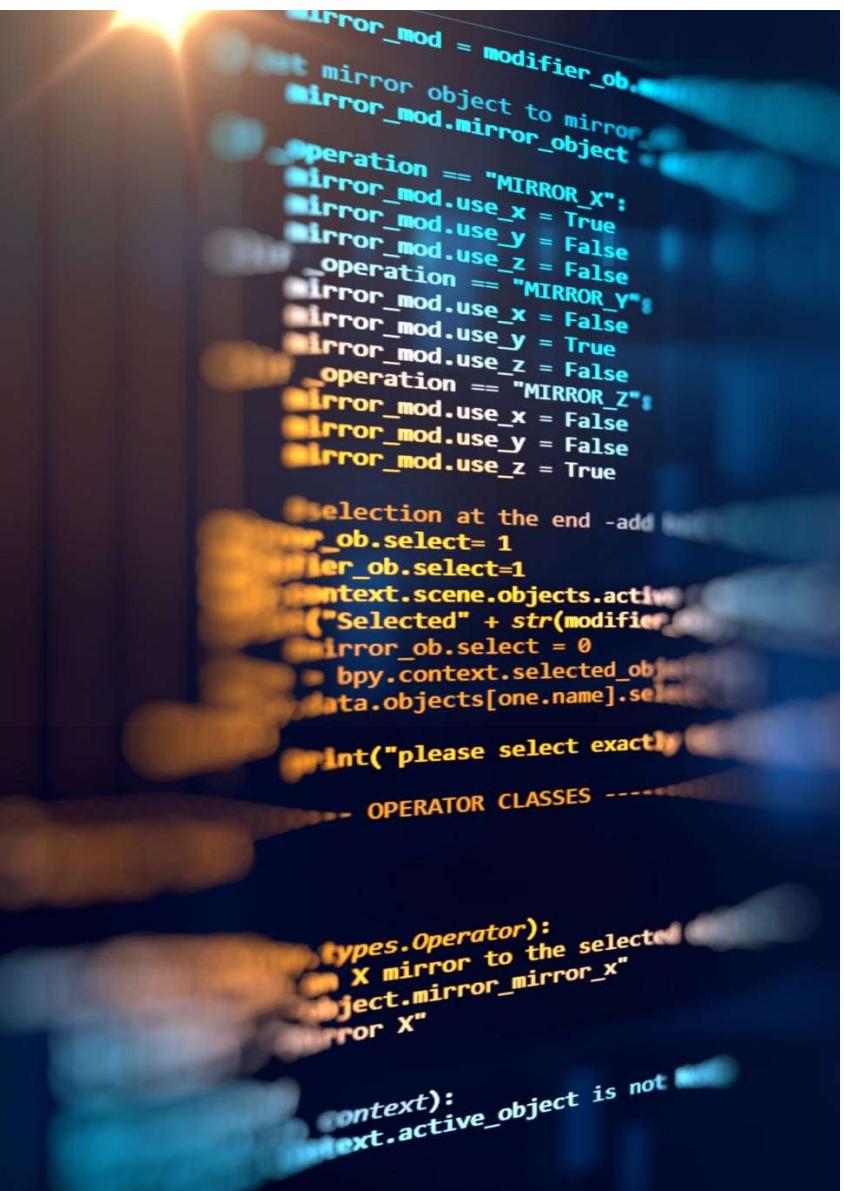
## Destroy a PHP Session

To remove all global session variables and destroy the session,  
use `session_unset()` and `session_destroy()`

```
<?php  
    .  
    // remove all session variables  
    session_unset();  
  
    // destroy the session  
    session_destroy();  
?>
```

# PHP Sessions Demo

- Review class DEMO 02a files



```
mirror_mod = modifier_ob.  
# mirror object to mirror  
# mirror_mod.mirror_object  
if operation == "MIRROR_X":  
    mirror_mod.use_x = True  
    mirror_mod.use_y = False  
    mirror_mod.use_z = False  
if operation == "MIRROR_Y":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = True  
    mirror_mod.use_z = False  
if operation == "MIRROR_Z":  
    mirror_mod.use_x = False  
    mirror_mod.use_y = False  
    mirror_mod.use_z = True  
  
# selection at the end -add  
# mirror_mod.select= 1  
# mirror_mod.select=1  
context.scene.objects.active = eval("Selected" + str(modifier))  
mirror_ob.select = 0  
bpy.context.selected_objects.append(mirror_mod)  
data.objects[one.name].select = 1  
  
print("please select exactly one object")  
  
-- OPERATOR CLASSES ---  
  
class MirrorOperator(bpy.types.Operator):  
    bl_idname = "object.mirror"  
    bl_label = "X mirror to the selected object"  
    bl_options = {'REGISTER', 'UNDO'}  
    bl_description = "mirror X"  
  
    def execute(self, context):  
        if context.active_object is not None:  
            ob = context.active_object
```