Załączanie plików

Włączanie kodu z innych plików do pliku głównego programu jest realizacją podejścia Reusability, czyli ponownego użycia kodu - w formie biblioteki, modułu, czy szablonu. Pozwala w sensowny sposób podzielić większy projekt na mniejsze, bardzie zrozumiałe części.

Załączanie plików skryptów PHP możliwe jest poprzez użycie jednej z 4 deklaracji (metod / instrukcja):

- include
- include_once
- require
- require_once

```
metoda(plik);
instrukcja plik;
```

Przykład:

```
include('rejestracja.php');
require 'logowanie.php';
```

Załączanie może zostać wywołane prawie w dowolnym miejscu skryptu (z zastrzeżeniem, aby w klasach nie wywoływać plików pomiędzy funkcjami), a nie tylko na początku pliku jak w przypadku np. C++ albo Javy.

Rożnice pomiędzy tymi 4 sposobami należy omówić w 2 kontekstach:
include vs. require oraz **include i require vs. include_once i require_once**

Include vs. Require

Pomimo, że obie instrukcje załączą wskazany plik to:

- include, jeżeli nie znajdzie wskazanego pliku, lub wystąpi błąd przetwarzania skryptu to i tak będzie kontynuował przetwarzanie skryptu, generując tylko ostrzeżenie, opcj. fn.
- require, jeżeli nie znajdzie wskazanego pliku, lub wystąpi błąd przetwarzania tego skryptu, to przerwie przetwarzanie skryptu zwracając błąd. Używane jeżeli bez danego skryptu aplikacja będzie niekompletna i źle działająca.

Include i Require vs Include_once i Require_once

Instrukcja ze słowem ONCE załączy wskazany plik tylko 1 raz, po uprzednim sprawdzeniu, czy wskazany plik został już załączony, podczas gdy instrukcja bez słowa once będzie załączać ponownie wskazany plik podczas uruchamiania skryptu tyle razy, ile interpreter znajdzie to polecenie w różnych skryptach podczas jednego przetwarzania.

Przykłady użycia:

- include – opcjonalne szablony
- include_once – opcjonalne zależności
- require – np. szablony, biblioteki
- require_once – zależności

AUTOLOADING

Przykład: include, czy include_once

Jest serwis składający się z różnych usług (każda usługa posiada własną klasę). Jest klient, który może korzystać z pojedynczej usługi, lub z usługi wywołującej po kolei kilka usług w jednym żądaniu. Każda usługa (klasa) wymaga tych samych bibliotek. Usługi można wywołać z kontrolera lub samodzielnie.

Jaką instrukcję wybrać, i gdzie umieścić załączanie wymaganych przez usługę bibliotek.

Przykład: include*, czy require*

Jest serwis podzielony na część publiczną o dostępie swobodnym oraz na część z dostępem dla zalogowanych użytkowników. Każda usługa z części nie-publicznej serwisu koniecznie wymaga biblioteki Autoryzującej dostęp, w przeciwnym przypadku nie można kontynuować pracy.

Ścieżki

Rozwiązywanie ścieżek do plików różni się pomiędzy stronami przetwarzania oraz używanymi językami. Inaczej można czytać adresy do zasobów w przeglądarce klienta, a inaczej w interpretatorze na serwerze mającym dostęp do całego dysku serwera.

Kwestie o których należy pamiętać:

- ścieżki relatywne i absolutne
- katalog dokumentów na Hostingu
- separatory ścieżki na systemach operacyjnych

Separatory w ścieżkach:

- system plików w Unix, Linux – znak: /
- Internet i serwery webowe – znak: /
- system MS Windows - znak \ (bieżący dysk) lub D:\ (dla całego systemu)

__FILE__ - pełna, absolutna ścieżka pliku aktualnie uruchomionego
__DIR__ - ścieżka z __FILE__ lecz bez nazwy pliku
realpath() pełna, kanoniczna i absolutna ścieżka dla wskazanego elementu
dirname() ścieżka dostępu do elementu wskazanego z prawej strony (końcówka stringu)

HDD: `/home/gp/Workspace/zse/TAI-K4/paths.php`
URL: `http://lo.local/zse/TAI-K4/paths.php`

```
__FILE__:
/home/gp/Workspace/zse/TAI-K4/paths.php

__DIR__:
/home/gp/Workspace/zse/TAI-K4

realpath():
/home/gp/Workspace/zse/TAI-K4

dirname():
/home/gp/Workspace/zse
```

HDD: `/home/gplwebpllk/edu/paths.php`
URL: `http://edu.gplweb.pl/paths.php`

```
__FILE__:
/home/gplwebpllk/edu/paths.php

__DIR__:
/home/gplwebpllk/edu

realpath():
/home/gplwebpllk/edu

dirname():
/home/gplwebpllk
```