Programování v PHP

Katedra softwarového inženýrství
Fakulta informačních technologií
České vysoké učení technické v Praze

Namespaces
BI-PHP

NAMESPACES umoznuji v ruznych include files ("knihovnach") pouzivat STEJNA jmena pro konstanty, funkce, tridy a interfaces

Byly definovany az v PHP 5.3

DEFINOVANI NAMESPACES:
- zadny kod pred namespace (s vyjimkou declare - kvuli encoding)
- PRED <?php zadne whitespaces ani ne-PHP kod (jako <html>)

namespaces se tykaji POUZE konstant, funkci, trid a interfaces
(i kdyz v namespace MUZE byt pouzit libovolny platny PHP kod)

namespaces mohou byt jednoduche: namespace jednoduchy;
nebo vnorene: namespace vnejsi\vnitrni;

lze vice namespaces v jednom souboru - nedoporucuje se
(lepe vice souboru, kazdy se svym namespace):

Syntaxe 1 - ns plati az do definice dalsiho ns:

```
    namespace prvni;

    // ...

    namespace druhy;

    // ...
```

Syntaxe 2 - rozsah ns urcen slozenymi zavorkami (NELZE VNOROVAT!!):
(vnorovani lze simulovat pomoci: namespace vnejsi\vnitrni;)

```
    namespace prvni{

       // ...

    }

    namespace druhy{

       // ...

    }
```

Pro kombinaci globalniho "beznamespacoveho" kodu (non-namespaced code)
s kodem v namespaces (namespaced code) lze pouzivat POUZE syntaxi se
slozeny zavorkami. Pritom ZADNY kod (s vyjimkou pripadneho declare)
nesmi byt mimo slozene zavorky!
namespaced a non-namespaced code:

```
    namespace pojmenovany{

       // ...

    }
    namespace { // globalni namespace - non-namespaced (zadne jmeno)

       // ...

    }
```

POUZIVANI NAMESPACES:
- unqualified names - jmena nepredchazena ns (neobsahujici \) - ("hola jmena")
- qualified names - jmena predchazena ns (obsahuji \), ale nezacinaji \ ("relativni cesty")
- fully qualified names - jmena predchazena \ ("absolutni cesty")

Neni-li pouzito use:
- pro unqualified a qualified se vzdy pripoji na zacatek aktualni ns
  pr.: je-li jmeno C\D (resp. X) a aktualni ns A\B, pouzije se A\B\C\D
  (resp. (A\B\X)

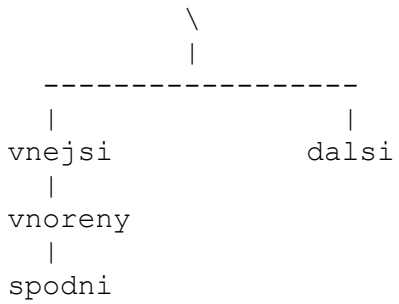
- unqualified functions - jsou vyhledavany take v globalnim ns (nejsou-li v aktualnim) - aby bylo mozno bezne pouzivat standardni fce

- pro fully qualified \A\B\C se pouzije beze zmeny

Pri pouziti use:
- je-li pouzito use A\B\C; (ekvivalent use A\B\C as C;)
  a jmeno je qualified (ne fully qualified) C\D\e(), dosadi se za alias C a vznikne A\B\C\D\e()

- je-li C jmeno unqualified class nebo interface a je pouzito use A\B\C as CLS; potom new CLS(); se vykona jako new A\B\C()

Struktura namespaces, pouzitych v prikladech:



Stranka, ktera includuje soubory, obsahujici namespaces:

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title></title>
    </head>
    <body>
        <?php
            include "ns_spodni.php";
            include "ns_vnoreny.php";
            include "ns_vnejsi.php";
            include "ns_dalsi.php";

            // definice funkci v souboru, kde neni definovan ns (globalni)
            function fce_v_namespace(){
                echo "Volana funkce 'fce_v_namespace' z globalniho ".
                     "namespace<br>";
            }

            function fce_v_globalnim(){
                echo "Volana funkce 'fce_v_globalnim' z globalniho ".
                     "namespace<br>";
            }

            // pouziti funkce, definovane v namespace i bez nej
            // nevznikne kolize - funkce v ruznych namespaces
            fce_v_namespace();          // totez \fce_v_namespace();
            vnejsi\fce_v_namespace();   // totez \vnejsi\fce_v_namespace();
            dalsi\fce_v_namespace();
            echo "<br>";

            // pouziti tridy, definovane v namespace
            vnejsi\class_v_namespace::Info();

            // pouziti konstanty, definovane v namespace
            echo vnejsi\const_v_namespace;

            // pouziti (implementace) interface, definovaneho v namespace
            class MyClass implements vnejsi\ISecurity_v_namespace{
                // ...

                public function Encode($key){
                    // ...
                    echo "volani fce Encode<br>";
                }

                public function Decode($key){
                    // ...

                }
            }
```

```
            // volani fce z vnoreneho namespace (jsme v globalnim ns - nic se
            // nedoplnuje)

            vnejsi\vnoreny\fce_z_vnoreneho();

            // volani funkce, ve ktere se volaji ruznym zpusobem
            // funkce z ruznych ns

            vnejsi\volej_fce_z_ns();

            // volani funkce, ve ktere se volaji ruznym zpusobem  funkce
            // za pouziti use

            vnejsi\volej_fce_pomoci_use();

            // volani funkce, ktera pouziva aliasy, definovane pomoci use

            vnejsi\volej_pomoci_aliasu();
            echo "<br>";

            // pouziti konstanty __namespace__
            echo "Aktualni namespace pro fci 'nacti_aktualni_namespace' je: ".
                 vnejsi\nacti_aktualni_namespace()."<br>";

            // pouziti klicoveho slova namespace
            vnejsi\pouzij_keyword_namespace();
        ?>
    </body>
</html>
```

Soubor ("knihovna") ns_vnejsi.php, ktery obsahuje vetsinu definic a pouzivani namespaces v techto prikladech

```php
<?php
    namespace vnejsi;

    function fce_v_namespace(){
        echo "Volana funkce 'fce_v_namespace' z namespace 'vnejsi'<br>";
    }

    // novejsi moznost definice konstanty
    const const_v_namespace = "Konstanta z namespace 'vnejsi'<br>";

    class class_v_namespace {
        // staticka metoda pouzita pro jednoduchost - aby se nemusela
        // vytvaret instance
        static function Info() {
            echo "Volana metoda 'Info' tridy 'class_v_namespace' z namespace ".
                 "'vnejsi'<br>";
        }
    }

    interface ISecurity_v_namespace{
        function Encode($key);
        function Decode($key);
    }


    // ruzna pouziti namespace pri volani funkci
    function volej_fce_z_ns(){
        echo "<br>--- Ruzna volani funkci z ruznych namespaces ---<br>";
        // automaticky se doplni (na zacatku) aktualni ns
        // (zde se vola \vnejsi\fce_v_namespace();)
        fce_v_namespace();

        // chyba - vola \vnejsi\fce_z_vnoreneho();
        // fce_z_vnoreneho();

        // OK - vola \vnejsi\vnoreny\fce_z_vnoreneho();
        // (opet doplni aktualni ns)
        vnoreny\fce_z_vnoreneho();

        // OK - pri "absolutni" (fully qualified - \ na zacatku)
        // se NEdoplnuje aktualni ns
        \vnejsi\vnoreny\fce_z_vnoreneho();

        // fully qualified (\...) NUTNE - jinak by doplnil aktualni ns
        \dalsi\fce_v_dalsi();

        // Pro NEkvalifikovane fce (a konstanty) hleda i v globalnim ns
           (nenajde-li je v aktualnim); NEPLATI pro classes!!
        // (umoznuje pouzivat standardni fce bez vedouciho \ )
        // ale pokud by stejnojmenna funkce (konstanta) byla v lokalnim
        // i globalnim ns, NUTNO volat takto: \fce_v_globalnim();
        fce_v_globalnim();
    }
```

```
    // Pouziti preddefinovaneho namespace:

    // "importovani" namespace (operator use); Lze i nekolik ns, oddelenych
    // carkami (popr. nekolik use)
    // funguje tak, ze vytvori alias: use vnejsi\vnoreny\spodni as spodni;
    // use NEZPUSOBI nejake prohledavani ns vnejsi\vnoreny\spodni
    //(na rozdil od C# apod.)!!
    use vnejsi\vnoreny\spodni;

    function volej_fce_pomoci_use(){
        echo "<br>--- Ruzna volani funkci s pouzitim use ---<br>";
        // chyba - unqualified (jen jmeno fce) - nepouzije use,
        // doplni a vola \vnejsi\fce_... :
        // fce_ze_spodniho();

        // fully qualified ("absolutni") - nedoplnuje, nepouziva use
        \vnejsi\vnoreny\spodni\fce_ze_spodniho();

        // chyba - doplni aktualni ns a vola: \vnejsi\vnejsi\...
        // vnejsi\vnoreny\spodni\fce_ze_spodniho();

        // OK - doplni, vznikne spravna "cesta" \vnejsi\vnoreny\spodni\fce_...
        vnoreny\spodni\fce_ze_spodniho();

        // OK - qualified - spodni nahradi "cestou"
        // z use (\vnejsi\vnoreny\spodni\fce_...)
        spodni\fce_ze_spodniho();
    }

    // pouziti use pro aliasy: vytvoreni zkratky (aliasu) pro dany ns
    use vnejsi\vnoreny\spodni as dolni;

    // vytvoreni zkratky (aliasu) pro TRIDU (nebo interface) v danem ns
    // (NELZE pro fce a const)
    use vnejsi\vnoreny\spodni\class_ze_spodniho as dolni_class;

    function volej_pomoci_aliasu(){
        echo "<br>--- Volani funkci a metod s pouzitim aliasu ".
             "vytvarenych use ---<br>";
        // dosadi za alias dolni (jiz nedoplnuje aktualni ns!)
        // za aliasem MUZE nasledovat cesta do vnorenych ns)
        dolni\fce_ze_spodniho();

        // opet dosadi za alias (class vcetne ns)
        dolni_class::Vypis();
    }

    // pouziti preddefinovane konstanty __namespace__
    function nacti_aktualni_namespace(){
        return __namespace__;
    }

    // pouziti klicoveho slova namespace:
    function pouzij_keyword_namespace(){
        // namespace lze pouzit k explicitnimu konstruovani
        // cesty od aktualniho ns
        return namespace\vnoreny\fce_z_vnoreneho()."<br>";
    }
?>
```

Soubor ns_vnoreny.php obsahujici funkci, definovanou ve vnorenem namespace:

```
<?php
    namespace vnejsi\vnoreny;

    function fce_z_vnoreneho(){
        echo "Volana funkce 'fce_z_vnoreneho' z namespace ".
             "'vnejsi\\vnoreny'<br>";
    }
?>
```

Soubor ns_spodni.php s jeste vice vnorenym namespace:

```
<?php
    namespace vnejsi\vnoreny\spodni;

    function fce_ze_spodniho(){
        echo "Volana funkce 'fce_ze_spodniho' z namespace ".
             "'vnejsi\\vnoreny\\spodni'<br>";
    }

    class class_ze_spodniho{
        static function Vypis(){
            echo "Volana metoda 'Vypis' tridy 'class_ze_spodniho' ".
                 "z namespace 'vnejsi\\vnoreny\\spodni'<br>";
        }
    }
?>
```

Soubor ns_dalsi.php jehoz namespace 'dalsi' NENI vnoren do ns vnejsi:

```
<?php
    namespace dalsi;


    function fce_v_namespace(){
        echo "Volana funkce 'fce_v_namespace' z namespace 'dalsi'<br>";
    }

    function fce_v_dalsi(){
       echo "Volana funkce 'fce_v_dalsi' z namespace 'dalsi'<br>";
    }
?>
```