INSTITUTO SUPERIOR TÉCNICO

PHP meets Ŭŋĭcøđ$

Nuno Lopes, NEIIST – 4º Ciclo de Apresentações, 13/Outubro/2005

Agenda:

- Porquê I10n/i18n?
- Desafios da I10n
- Introdução ao Unicode
- Implementação Actual (PHP 4/5)
- Implementação Futura (PHP 6)

- Links
- Questões

Agenda:

- ⇨ **Porquê I10n/i18n?**
- Desafios da I10n
- Introdução ao Unicode
- Implementação Actual (PHP 4/5)
- Implementação Futura (PHP 6)

- Links
- Questões

Porquê I10n/i18n?

- There is more than one country in the world
- Ce n’est pas tout le monde qui parle anglais
- Tjueseks karakterer holder ikke mål
- Нот эврибади из юзин зэ сэйм скрипт ивэн
- 它变得更加复杂的与汉语语言

Porquê I10n/i18n?

- Suportar as línguas necessárias, sem rescrever a aplicação
- Adicionar novos caracteres de forma transparente (por exemplo, €)

Agenda:

- Porquê I10n/i18n?
- ⇨ **Desafios da I10n**
- Introdução ao Unicode
- Implementação Actual (PHP 4/5)
- Implementação Futura (PHP 6)

- Links
- Questões

Desafios da I10n

- Diferenças nos charsets
- Multi-byte vs Single-byte encodings
- Diferentes algoritmos de sort, spelling, dates, ...

Exemplo: Sorting (aka Collation)

- Em Lituano, o ‘*y*’ é ordenado entre ‘*i*’ e ‘*k*’
- Em Espanhol Tradicional, ‘*ch*’ é tratado como uma única letra, e é ordenado entre ‘*c*’ e ‘*d*’
- Em Sueco, ‘*v*’ e ‘*w*’ são consideradas variantes da mesma letra
- Em Alemão, ‘*öf*’ é ordenado antes de ‘*of*’. Nas listas telefónicas é o contrário

Exemplo: Capitalization

- Grego: Σ ⇨ σ (no meio de uma palavra)
- Grego: Σ ⇨ ς (no fim de uma palavra)
- Turco: i ⇨ İ, ı ⇨ I
- Alemão: ß ⇨ SS (lower[SS]=ss)

Agenda:

- Porquê I10n/i18n?
- Desafios da I10n
- ⇨ **Introdução ao Unicode**
- Implementação Actual (PHP 4/5)
- Implementação Futura (PHP 6)

- Links
- Questões

Introdução ao Unicode

- Suporta todas as línguas
- +100 mil caracteres
- 1 caracter != 1 byte
- Compatível com ASCII
- BOM (byte order mask) identifica a codificação usada

Termos técnicos (UTF-16)

- Code point – representação de caracteres por números (U+1234)
- Code unit – uma sequência de dois bytes
- Surrogates (high and low) – 2 code units para representar o mesmo caracter (> FFFF)

Codificação

- UTF-7 (obsoleto)
- UTF-8 (até 4 bytes)
- UTF-16 (LE & BE) (2 ou 4 bytes)
- UTF-32 (LE & BE) (4 bytes)
- UTF-EBCDIC (até 5 bytes)
- ...

Composição de caracteres

a + ˆ + . = ậ

U+0061 + U+0302 + U+0323 = U+1EAD

a + . + ˆ = ậ

U+0061 + U+0323 + U+0302 = U+1EAD

Normalization

- Caracteres equivalentes são reduzidos a uma forma standard (por exemplo os caracteres do ASCII estendido)
- Facilita algoritmos

å != å

U+00C5 + U+030A != U+0041

Propriedades

- Os caracteres têm propriedades, como:
  - Espaços
  - Letras (lower/upper case)
  - Números
  - Pontuação
  - ...

Agenda:

- Porquê I10n/i18n?
- Desafios da I10n
- Introdução ao Unicode
- ⇨ **Implementação Actual (PHP 4/5)**
- Implementação Futura (PHP 6)

- Links
- Questões

Iconv

- iconv_strlen()
- iconv_substr()
- iconv_strpos()
- iconv()

- Não resolve a maioria dos problemas

Mbstring

- mb_strlen()
- mb_strpos()
- ...

- Centrado em charsets Asiáticos
- Também não resolve a maioria dos problemas

Agenda:

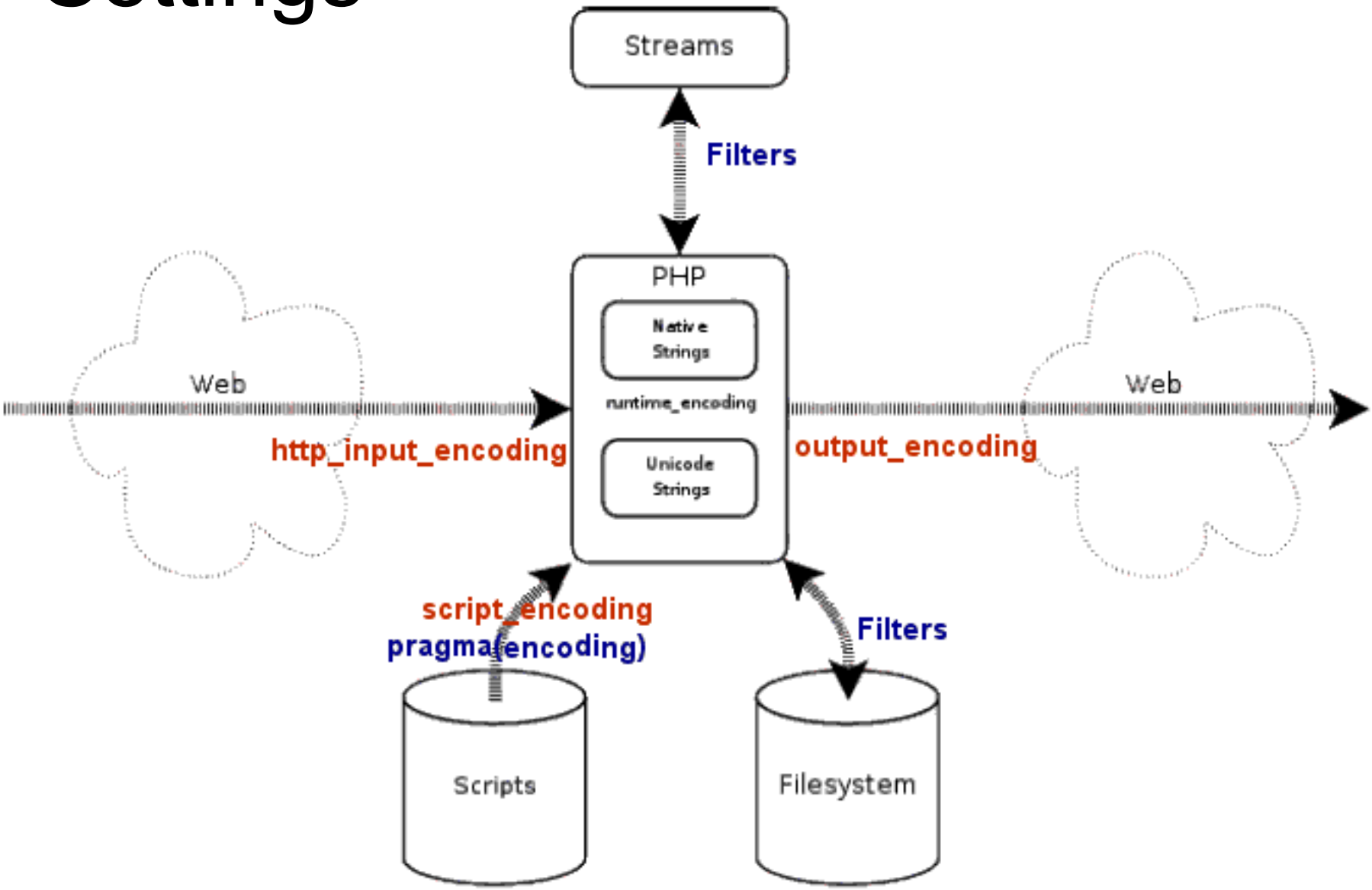
- Porquê I10n/i18n?
- Desafios da I10n
- Introdução ao Unicode
- Implementação Actual (PHP 4/5)
- ⇨ **Implementação Futura (PHP 6)**

- Links
- Questões

PHP 6

- Detecção do encoding do script via BOM
- Overload das funções de forma transparente
- Variáveis e nomes de funções em Unicode
- Suporte para Locales POSIX
- Utiliza a library da IBM: ICU
- UTF-16 internamente

Settings



Hello World

```
<?php

ini_set('unicode.output_encoding', 'iso-8859-1');

function こんにちは () {
    $世界 = 'Hello World!';

    echo $世界 ;
}

こんにちは ();

?>
```

Sorting

```php
<?php

// the list of the strings to sort
$array = array(
   'caramelo',
   'cacto',
   'caçada'
);

// set our locale (Portuguese, in this case)
i18n_loc_set_default('pt_PT');

// sort using the locale we previously set
sort($array, SORT_LOCALE_STRING);

?>
```

Normalization

```
<?php

$GLOBALS["\u212B"] = ' 승인 ';

// U+00C5 = Å

echo $GLOBALS["\u00C5"];


?>
```

String types

- binary – raw strings
- string – usa o encoding do script (for BC)
- unicode – UTF-16

Binary vs Unicode

```
<?php

$unicode = ' 傀儡两亨メ了乂刃 ';

$binary  = b' 傀儡两亨メ了乂刃 ';
$binary2 = (binary) $unicode;

echo strlen($unicode); // 8
echo strlen($binary);  // 24
echo strlen($binary2); // 24

var_inspect($unicode[2]); // unicode(1) " 两 " { 4e24 }
var_dump($binary[2]); // Ç

?>
```

Escapes

```
<?php

// '\Uxxxxxx'
$str = 'U+123: \U000123';

// '\uxxxx'
$str = 'U+123: \u0123';



// unicode(8) "U+123: ģ"
var_dump($str);

?>
```

Novas funções

- unicode **unicode_decode**(input, encoding)
- string **unicode_encode**(input, encoding)
- string **i18n_loc_get_default**()
- bool **i18n_loc_set_default**(locale)
- text **i18n_strtotitle**(str)
- ...?

Stream Filters

- unicode.to.* - Unicode->String
- unicode.from.* - String->Unicode
- unicode.tidy.* - “magic” filter

Agenda:

- Porquê I10n/i18n?
- Desafios da I10n
- Introdução ao Unicode
- Implementação Actual (PHP 4/5)
- Implementação Futura (PHP 6)

⇨ **Links**

- Questões

Links

- www.php.net/unicode
- http://www.derickrethans.nl/files/php6-unicode.pdf
- http://www.gravitonic.com/do_download.php?dow
- http://mega.ist.utl.pt/~ncpl/pres/

INSTITUTO SUPERIOR TÉCNICO

PHP meets Ŭŋĭcøđ∑