Desenvolvimento Aberto

Localização e internacionalização de software

Igor dos Santos Montagner (igorsm1@insper.edu.br)

Avisos

- Este curso exige e exercita autonomia.
    - Não serão mais cobradas as skills de sala de aula
    - Não **enviar** skills implica em reprovação
    - PRs vindos do `master` serão negados
- Algumas skills tem data para podermos discutir experiências e resultados
- Não existem penalizações no curso
    - mas elas podem eventualmente ser criadas

Tradução de software

Qual a diferença de localização e internacionalização?

Internacionalização (I18N)

- Consiste em traduzir a interface de usuário de um software para outros idiomas.
- SO guarda configurações de idioma e as disponibiliza para aplicações
- Tipicamente "invisível"

Internacionalização (I18N)

(exemplo com okular)

Localização (L10N)

Consiste em adaptar a maneira de mostrar

- números fracionários
  - marcador de decimais
  - marcador de milhares
- datas
  - nomes de meses
  - ordem de exibição
- nomes de países, fusos horários, etc

de acordo com as preferências de um usuário e de sua cultura.

L10N vs I18N

Precisam ser

- independentes:
  - idioma inglês e datas no formato brasileiro
- configuráveis
  - posso precisar trocar entre línguas

O suporte a L10N e I18N implica modificar código fonte.

Locales

Um *locale* é uma tripla

```
<lingua>_<pais>.<codificacao>
```

que representa configurações de I18N e/ou L10N para uma determinada cultura.

Locales

Exemplos:

- Tradução de *File*:
  - `pt` = Ficheiro
  - `pt_BR` = Arquivo
- Formato de datas:
  - `en_US`: *MM/DD/YY*
  - `en_GB`: *DD/MM/YY*

Posso usar *locales* diferentes para a língua da interface de usuário e para mostrar datas.

Suporte a L10N

1. Baixar uma biblioteca de Localização
2. Encontrar todas as exibições de números, datas, etc
3. Pré-processá-las usando funções da biblioteca

> print(10.5) --> print(format_number(10.5))

Não é complicado, mas é **trabalhoso**

Suporte a I18N

Envolve 3 etapas:

1. Marcar todas strings que devem ser traduzidas
2. Extraí-las do código fonte
3. Criar um arquivo de traduções para cada *locale* suportado
4. Empacoter as traduções junto com o programa

É um pouco mais complicado, mas pode ser integrado ao processo de compilação de um programa.

Suporte a I18N (POSIX)

Sistemas POSIX suportam determinação de lingua e locale usando variáveis de ambiente.

- `LANG` para língua
- `LC_TIME` para data
- `LC_NUMERIC` para números

Um locale sempre é expresso no formato

```
<lingua>_<pais>.<codificacao>
```

Suporte a I18N (POSIX)

O comando `locale` mostra todas as opções disponíveis:

```
LANG=en_US.UTF-8
LANGUAGE=
LC_CTYPE=pt_BR.UTF-8
LC_NUMERIC=pt_BR.UTF-8
LC_TIME=pt_BR.UTF-8
LC_COLLATE="en_US.UTF-8"
LC_MONETARY=pt_BR.UTF-8
LC_MESSAGES="en_US.UTF-8"
LC_PAPER=pt_BR.UTF-8
LC_NAME=pt_BR.UTF-8
LC_ADDRESS=pt_BR.UTF-8
LC_TELEPHONE=pt_BR.UTF-8
LC_MEASUREMENT=pt_BR.UTF-8
LC_IDENTIFICATION=pt_BR.UTF-8
LC_ALL=
```

Suporte a I18N (Web)

Existem diversas maneiras de determinar um bom locale em sistemas Web:

- Cabeçalho HTTP `Accept-Language` inclui as linguagens de exibição suportadas pelo browser do visitante.
- Geolocalização via IP
- Preferência armazenada em banco de dados

Web e desktop usam as mesmas tecnologias (l10n e i18n)

Suporte a I18N (em Python)

- Módulo `gettext` da biblioteca padrão
- Módulo `datetime` aceita o uso de locales
- Módulo `babel` que adiciona facilidades de localização e facilita o fluxo de criação das traduções.

Atividade prática

Usaremos o módulo *Babel* para traduzir uma aplicação do terminal.

- **skill**: *Tradução básica*

O handout também já tem instruções para a primeira skill obrigatória para o **C**:

- **skill**: *Tradução aceita!*

Desenvolvimento Aberto

Localização e internacionalização de software

Igor dos Santos Montagner (igorsm1@insper.edu.br)