Processando eventos de teclas independente de plataforma

Descrição do problema

Enquanto desenvolvem games para vários dipositivos habilitados com J2ME/MIDP, os desenvolvedores frequentemente dão de cara com o problema de identificação das teclas pressionadas. Isto acontece pela razão que os diferentes fabricantes de aparelhos usam diferentes códidos de tecla (key-codes) para as memas teclas.

Alguma teclas podem ser definidas com o método *javax.microedition.lcdui.Canvas.getGameAction(int)*. Mas algumas não podem, por exemplo: erase key, back-key nos SonyEricsson etc.

Este problema é redobrado no caso de arquivos JAR de games destinados a 'rodar' em aparelhos de diferentes fabricantes.

Solução

Eu sugiro a seguinte solução:

- Crie uma classe KeyCodeAdapter separadamente.
- Defina um conjunto interno de constantes nesta class. Estas constantes serão usadas dentro do game/aplicação.
- Defina conjutos de constantes para cada fabricante de aparelho, para os quais desejamos dar suporte no game/aplicação.
- Agora devemos determinar o fabricante do aparelho onde a aplicação foi iniciada. Para isso podemos usar o código a seguir:

```
private String getPlatform() {
        // detectando NOKIA ou SonyEricsson
        try {
            final String currentPlatform = System.getProperty(
                                        "microedition.platform");
            if ( currentPlatform.indexOf( "Nokia") != -1) {
                return PLATFORM_NOKIA;
            } else if ( currentPlatform.indexOf(
                                        "SonyEricsson") != -1) {
                return PLATFORM_SONY_ERICSSON;
            }
        } catch ( Throwable ex) {
        }
        // detectando SAMSUNG
        try {
            Class.forName( "com.samsung.util.Vibration");
            return PLATFORM_SAMSUNG;
        } catch ( Throwable ex) {
        }
        // detectando MOTOROLA
        try {
            Class.forName( "com.motorola.multimedia.Vibrator");
            return PLATFORM_MOTOROLA;
        } catch ( Throwable ex) {
            try {
                Class.forName( "com.motorola.graphics.j3d.Effect3D");
                return PLATFORM_MOTOROLA;
            } catch ( Throwable ex2) {
                try {
                    Class.forName( "com.motorola.multimedia.Lighting");
                    return PLATFORM_MOTOROLA;
                } catch ( Throwable ex3) {
                    try {
                        Class.forName( "com.motorola.multimedia.FunLight");
                        return PLATFORM_MOTOROLA;
                    } catch ( Throwable ex4) {
                    }
                }
            }
        }
        try {
            if ( adaptorCanvas.getKeyName( SOFT_KEY_LEFT_MOTOROLA
                        ).toUpperCase().indexOf(SOFT_WORD) > -1) {
                return PLATFORM_MOTOROLA;
            }
        } catch ( Throwable e) {
            try {
                if ( adaptorCanvas.getKeyName( SOFT_KEY_LEFT_MOTOROLA1
                        ).toUpperCase().indexOf(SOFT_WORD) > -1) {
                    return PLATFORM_MOTOROLA;
                }
            } catch ( Throwable e1) {
                try {
                    if ( adaptorCanvas.getKeyName( SOFT_KEY_LEFT_MOTOROLA2
                        ).toUpperCase().indexOf(SOFT_WORD) > -1) {
                        return PLATFORM_MOTOROLA;
                    }
                } catch ( Throwable e2) {
                }
            }
        }
        // detectando SIEMENS
        try {
            Class.forName( "com.siemens.mp.io.File");
            return PLATFORM_SIEMENS;
        } catch ( Throwable ex) {
        }
        // detectando LG
        try {
            Class.forName( "mmpp.media.MediaPlayer");
```


(C) Copyright Nokia 2013. All rights reserved.

```
            return PLATFORM_LG;
        } catch ( Throwable ex) {
            try {
                Class.forName( "mmpp.phone.Phone");
                return PLATFORM_LG;
            } catch ( Throwable ex1) {
                try {
                    Class.forName( "mmpp.lang.MathFP");
                    return PLATFORM_LG;
                } catch ( Throwable ex2) {
                    try {
                        Class.forName( "mmpp.media.BackLight");
                        return PLATFORM_LG;
                    } catch ( Throwable ex3) {
                    }
                }
            }
        }
        return PLATFORM_NOT_DEFINED;
    }
```

O método acima retona uma das constantes que contem o nome da plataforma. Podemos chamar este médodo no construtor de nossa class KeyCodeAdapter e fazer desta classe um sigletone. Desta maneira não precisamos nos preocupar em ficar chamando o método getPlatform() manualmente, porque este será chamado automaticamente após cada uso de nossa class.

Exemplo de uso

Seguindo as ações descritas acima é o suficiente para transformar os códigos de tecla (key-codes) nativos, fornecidos pelo Canvas em nossos códigos. Isto pode ser feito fazendo uma simples chamada ao método adoptKeyCode(), como a seguir:

```
    protected void keyPressed(int keyCode) {
        int internalKeyCode =
            KeyCodeAdapter.getInstance().adoptKeyCode(keyCode);
        switch (keyCode) {
            case KeyCodeAdapter.SOFT_KEY_LEFT:
                // algum processamento
                break;
            case KeyCodeAdapter.BACK_KEY:
                // algum processamento
                break;
            default:
        }
        ...
    }
```

Sumário

- O código completo pode ser baixado aqui: File:KeyCodeAdapter.zip**.**

- Esta classe também pode ser usada para determinar os fabricantes do aparelho onde o game/aplicação está sendo iniciado, simplesmente chamando o método getPlatform.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.