

# Especificación de juegos Arcade 1.0

Jurgen Heysen

January 6, 2014

## Contents

## *CONTENTS*

<b>1</b>	<b>Introducción</b>	<b>3</b>
<b>2</b>	<b>Estándar</b>	<b>3</b>
2.1	Código . . . . .	3
2.2	XML . . . . .	3
<b>3</b>	<b>Servicios</b>	<b>4</b>
3.1	Clase ArcadeServiceInterface . . . . .	4
3.1.1	Método getServices() . . . . .	4
3.1.2	Método getService(servicio) . . . . .	4
<b>4</b>	<b>Instalación de juegos</b>	<b>4</b>

## 1 Introducción

El presente documento tiene como objetivo instruir a los desarrolladores de videojuegos sobre los estándares que deben cumplir para ser compatibles con el Launcher del Arcade

## 2 Estándar

### 2.1 Código

Los juegos deben exponer un archivo .py con una clase que contenga el método `Go`, que debe recibir como parámetro un objeto del tipo `ArcadeServiceInterface` (`ArcadeServiceInterface.py`), este objeto será tratado mas adelante y es una interfaz a los servicios solicitados por el juego.

### 2.2 XML

Los juegos deben exponer un archivo .xml con información relevante sobre ellos. El nodo principal debe llamarse `<Game>` y opcionalmente de argumento puede llevar un string llamado `version` con el valor "1.0". El Título del juego va entre las etiquetas `<Title>` y `</Title>`. Es importante agregar un código corto único para el juego entre las etiquetas `<Code>` y `</Code>`.

Las etiquetas `<Description>` y `</Description>` encierran la descripción del juego.

La versión se incluye entre `<Version>` y `</Version>`, se recomienda que sea un entero

La fecha de compilación puede ser añadida con `<Date>` y `</Date>`

La etiqueta `<MainClass..>` es obligatoria y debe llevar como argumentos `name` con el nombre de la clase que contiene el método `Go(...)` y `path` con la ruta relativa a este archivo, extensión incluida.

Se puede incluir la lista de autores mediante una sección `<Authors>` finalizada con `</Authors>`. Cada autor se individualiza `<Author name="..." />`, donde `name` es el nombre del autor.

Otro aspecto a considerar es la lista contenida entre `<Screenshots>` y `</Screenshots>` para incluir las screenshots del juego que se mostraran en el Launcher. cada una se indica con la etiqueta `<Screenshot src="..." />` donde `src` es el path a la screenshot.

La lista denotada por `<Services>` y `</Services>` indica los servicios que solicita el juego al Launcher, cada servicio se solicita mediante `<Service name="..." />` donde `name` es el nombre del servicio solicitado, además se deben incluir los parámetros requeridos por el servicio según su propio estándar.

Adicionalmente se puede crear una lista denotada por `<AdditionalData>` y `</AdditionalData>` para incluir datos adicionales. No hay especificaciones sobre los contenidos de esta lista.

A continuación se presenta una tabla resumen sobre la información que debe contener el xml:

Table 1: Resumen de documento Xml

Tag	Parent	Args	Num	Descripción
Game	-	-	1	Root del documento
Title	Game	-	1	Título del juego
Code	Game	-	1	Código del juego
MainClass	Game	name: Nombre file: archivo	1	Nombre de la clase de inicialización del juego
Description	Game	-	1	Descripción
Version	Game	-	1	Versión del juego
Date	Game	-	1	Fecha de release
Authors	Game	-	1	Encierra lista de Autores
Author	Authors	name: Nombre	*	Identifica un autor
Screenshots	Game	-	1	Encierra lista de capturas de pantalla
Screenshot	Screenshots	src: path a la imagen	*	Identifica una captura de pantalla
Services	Game	-	1	Encierra lista de servicios
Service	Services	name: Nombre del servicio parámetros adicionales del servicio	*	Identifica un servicio que solicita el juego
AdditionalData	Game	-	1	Encierra una lista de datos adicionales

## 3 Servicios

El Launcher ofrece a los juegos diversos servicios que pueden ser instalados. Estos son apegados a la instalación base y siguiendo la documentación de servicios, los desarrolladores pueden incluir los suyos propios.

En esta sección se abordará la interfaz común de acceso a servicios.

### 3.1 Clase `ArcadeServiceInterface`

La clase `ArcadeServiceInterface` provee el acceso a los servicios del Arcade que son declarados en el .xml de información del juego. Los servicios contenidos en él se encuentran ya inicializados para el juego, se provee dos formas de acceder a los servicios.

#### 3.1.1 Método `getServices()`

Este método devuelve un diccionario con las clases que representan cada servicio, donde las llaves son los nombres de estos. Si faltara algún servicio, significa que este no se registró correctamente en la inicialización o que no se encuentra instalado, pero para todos los efectos no está disponible.

#### 3.1.2 Método `getService(servicio)`

Este método recibe como string el nombre del servicio y retorna el objeto que representa a este servicio si se encuentra disponible o `None` en caso contrario.

## 4 Instalación de juegos

En una distribución normal del Launcher, los juegos pueden ser instalados siguiendo los siguientes pasos:

1. Descomprimir los contenidos del juego en una carpeta de elección.
2. Navegar a carpeta del Launcher.
3. En consola de comandos, ejecutar `python game_install.py -i path_a_xml` <sup>1</sup>
4. Verificar con `python game_install.py -l`

Puede desinstalar<sup>2</sup> un juego con `python game_install.py -u CODIGO` donde CODIGO es el código del juego que desea desinstalar. Puede ser visto ejecutando antes `python game_install.py -l`

---

<sup>1</sup>Ajustar a python 2.7 si no fuera el caso

<sup>2</sup>No borra los archivos, solo quita el juego del registro del arcade