
GAMELOFT

MOBILE GAME PROGRAMMING

OUTLINE

- Production circle
- Porting! Why and What?
- Game Production Process
- Development Tools
- Package & Release
- Some simple Optimization Tricks

Production Circle

GRAPHIC ARTISTS
Pixel Art

- *Character design*
- *Animation*
- *Interface design*
- *Backgrounds*
- *Mapping*

GAME DESIGNER
Idea ($!)
Design

- *Game play*
- *Story, character*
- *Level*

PRODUCER
Project Management

- *Task Breakdown*
- *Priority*
- *Schedule*
- *HR allocation*

PROGRAMMER
Coding

- *Coding*
- *Scripting*
- *Debugging*

TESTERS
Quality Control

- *Testing*
- *Bugs report*

GAMELOFT

PORTING! WHY AND WHAT?

Porting! Why and What?

Touch?

Create on k800i

Or Noki And…

- **Size:** 240x320
- **Mem:** 4M

Run OK?

: 128x160

: ~500K

Size: 240x320

mobile?

Size: 320x240

Mem: 7M

Brew platform?

GAMELOFT

Porting! Why and What?

Creation

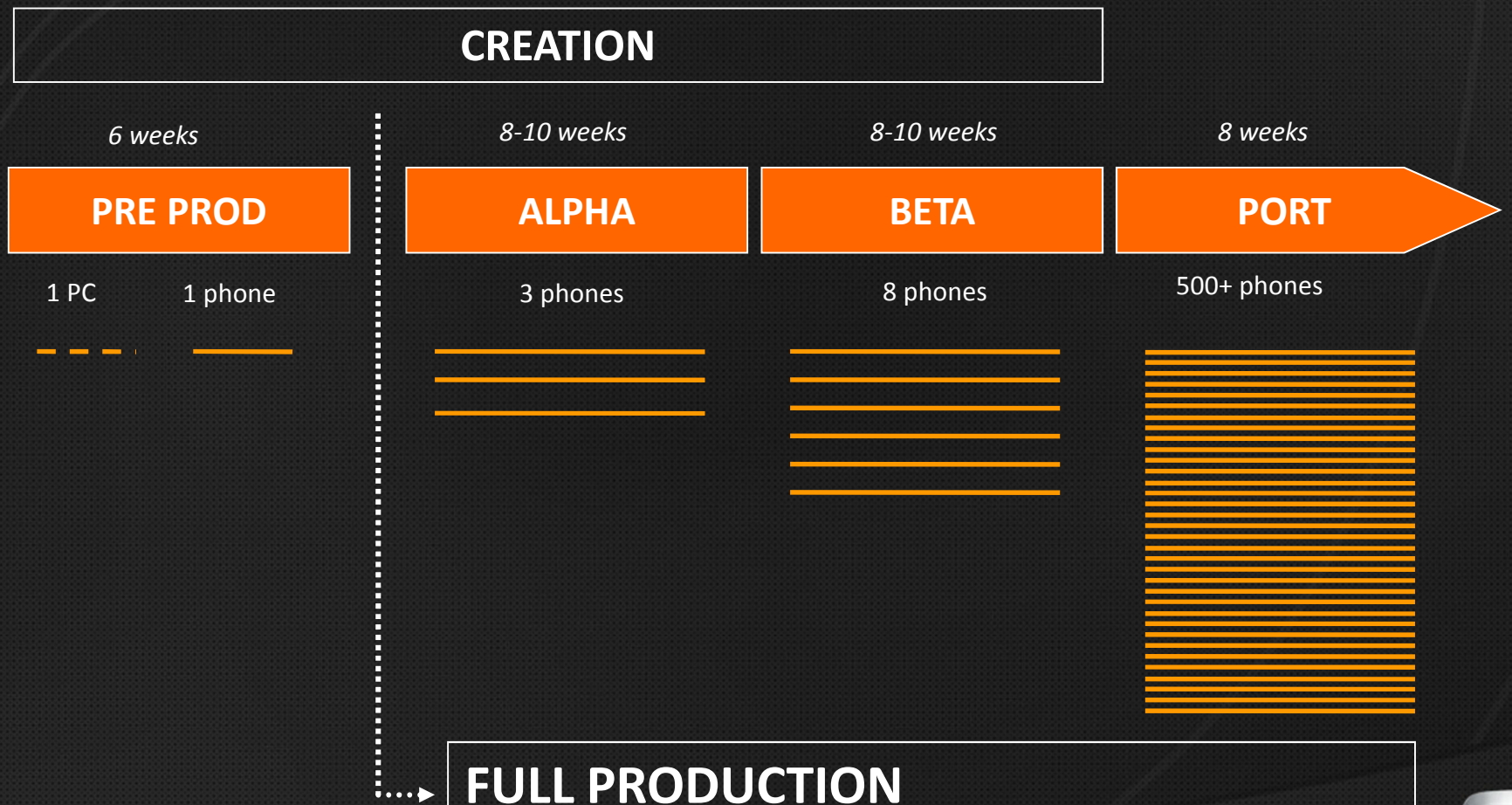
- Create and develop **new** games

Porting

- **Adapt** games to over 500 devices/platforms (low-end to high-end), and for different international markets (Europe, America, Asia…)

GAME PRODUCTION PROCESS

Production Process

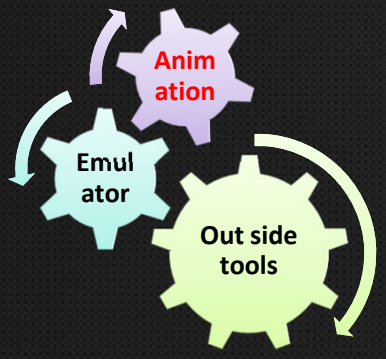


GAMELOFT

DEVELOPMENT TOOLS

Development Tools

- Inside Tools
  - Animation Manager
  - Emulator
- Outside Tools
  - CPP: C Preprocessor
  - Proguard: optimizing, obfuscating, shrinking,..

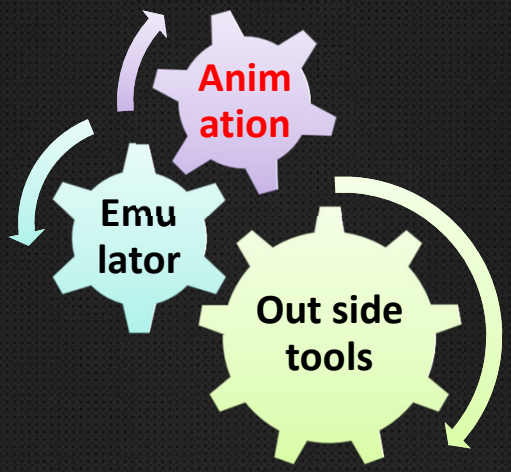


GAMELOFT

Development Tools

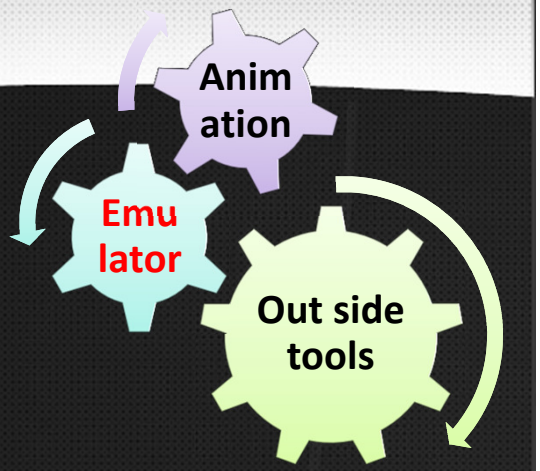
- Awesome, convenient, realistic!

Demo video about this tool!



GAMELOFT

Development Tools



- Powerful, effective and visual!

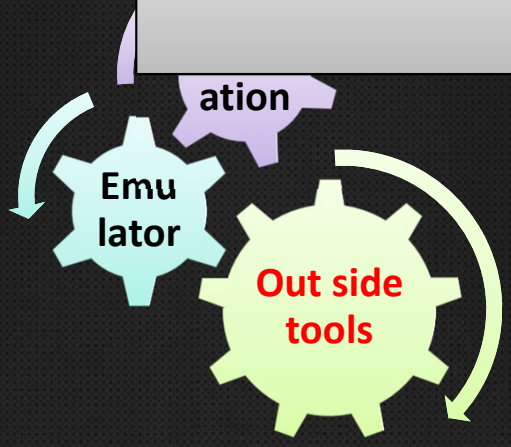
Demo video about this tool!

GAMELOFT

Development Tools

- Preprocess (CPP)

How C++ preprocess can be written on Java code?



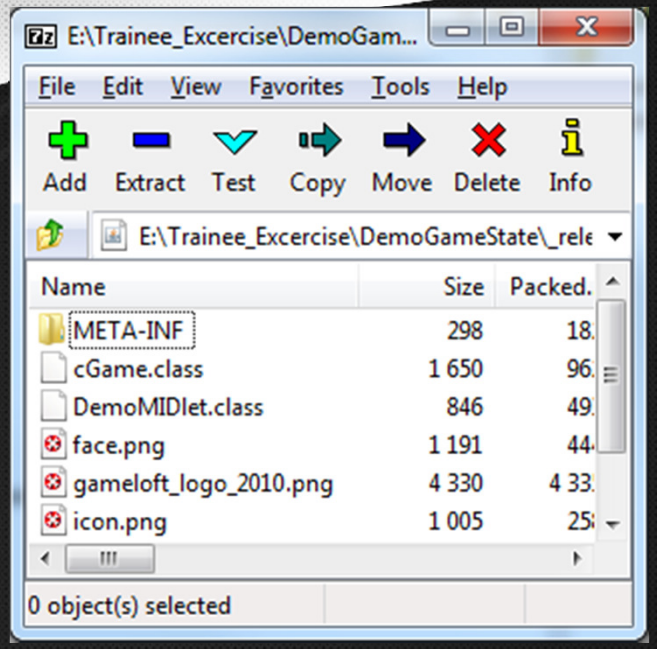

```
public class cGame extends Canvas implements Runnable
{
        #include "defines.h"

#if SONY_K800i
        static final int SCREEN_W = 240;
        static final int SCREEN_H = 320;
#else
        static final int SCREEN_W = SCR_W;
        static final int SCREEN_H = SCR_H;
#endif

        public void paint(Graphics g)
        {
                                                        ;
}
```

Have you ever seen this?

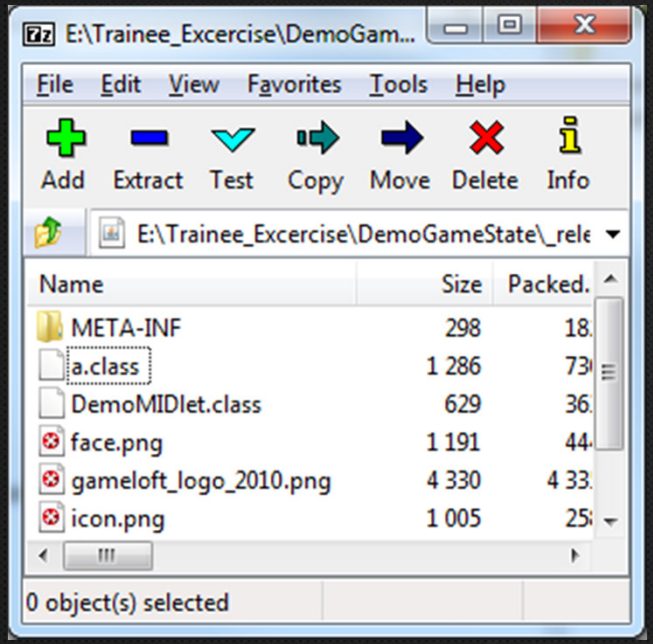
"Cpp.exe" -C -P -DSCR_W=480 Input\MyCanvas.java Output\MyCanvas.java

PACKAGE & RELEASE

Package & Release



?



What are the differences?

GAMELOFT

SOME OPTIMIZATION TRICKS

Some Optimization Tricks

- Optimizing Memory
  - Image
    - Cache/not cache.
    - Split Image into small Image.
  - Sound
  - String vs byte array
- Optimizing Performance
  - High level
  - Low level

Cache or not?

All images are created in Loading or Initializing step

Load all data to a byte array

I am phone memory

Byte[]

cache

not cache

Cost more memory but faster

createImage(...)

drawImage(image,...)

drawRGB(int[],...)

Saving memory but slower

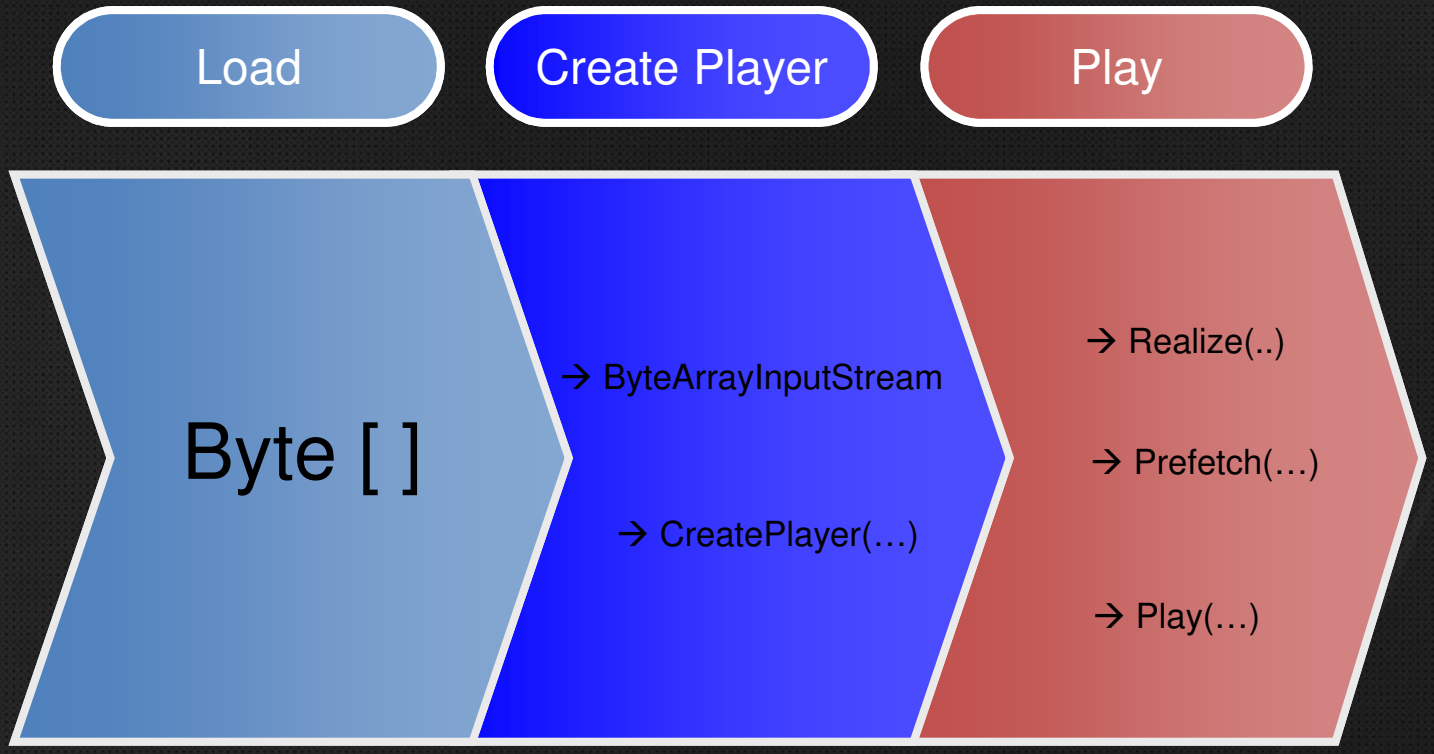
GAMELOFT

Split Image into small Image

GAMELOFT

Optimizing Memory > Sound

To play a sound:



byte[] < ByteArrayInputStream < Player

→ Sound data is much larger when decompressed

GAMELOFT

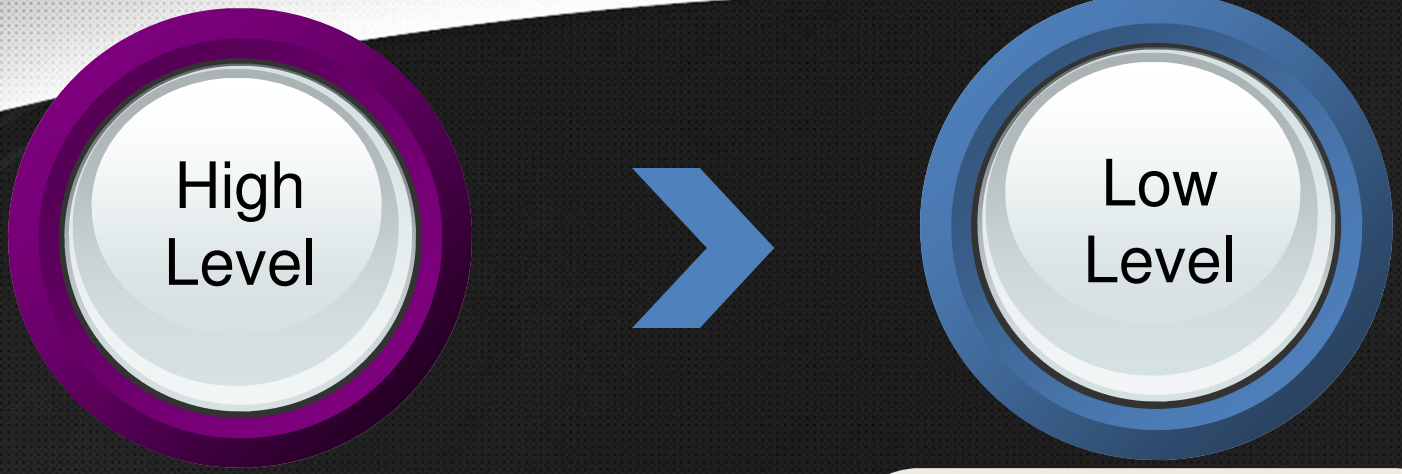
Optimizing Memory > String vs Byte Array

Normally, we use String object to store text.

String	Byte arrays
✓ Class header.	✓ *No* class header.
✓ Methods and Private member variables.	✓ *No* methods and private member variables.
	✓ *Encoded*

→ String consumes more memory than byte array.

Optimizing Performance



-Optimize algorithms.
- Difficult and specific for each game
- Provides a great improvement.

- Optimize calculations and execution conditions.
- Can be applied to all games
- Provides only a little improvement.

→ As a rule, *High-level* optimizations should be considered before doing *Low-level* optimizations.

GAMELOFT

Optimizing Performance

High Level

- Buffer technique
  - Back buffer:
  - Circular buffer:
  - …

Optimizing Performance

Low Level

- This code snippet below will vertically flip a RGB array of a 256x256 image. What problem?

```
int i = 0;
while (i < 256)
{
    int j = 0;
    int tmp =  i << 8 ;
    while (j < 256)
    {
        m_aiDestPixels[ i*256 + j] = m_aiSrcPixels[ i*256  + (256 - j)];
        j = j + 1;
    }
    i = i + 1;
}
```

GAMELOFT

Optimizing Performance

- This code snippet below will vertically flip a RGB array of a 256x256 image. What problem? **Low Level**

```
int i = 0;
while ( i <= 256)          <-- Compare to 0 is faster
{
    int j = 0;
    int tmp = i << 8 ;
    while ( j <= 256)      <-- Compare to 0 is faster
    {
        m_aiDestPixels[ tmp + j] = m_aiSrcPixels[ tmp + (256 - j)];

    } j += j + 1;          <-- Use ++ is faster

} i += i + 1;              <-- Use ++ is faster
```

SUMMARY

- Porting! Why and What?
- Game Production Process
- Development Tools
- Package & Deploy
- Some simple Optimization Tricks

Mobile
Game Designer

Game Features

Game feature is what the game has

- Story
- Levels
- Effect
- Rules
- Skill
- ...

How to make an interesting game

- Balance
- Story
- Choices
- Idea
- User

GAMELOFT

THANK YOU