Contents

[Development goals 3](#_Toc257028685)

[Jobs (task types) 3](#_Toc257028686)

[Tools 3](#_Toc257028687)

[Game overview 3](#_Toc257028688)

[Prototypes 4](#_Toc257028689)

[System requirements 9](#_Toc257028690)

[Functional description 10](#_Toc257028691)

[Game screens 10](#_Toc257028692)

[Loading 10](#_Toc257028693)

[Logo 10](#_Toc257028694)

[Menu 10](#_Toc257028695)

[Credits 11](#_Toc257028696)

[Level selection 12](#_Toc257028697)

[Game 12](#_Toc257028698)

[Game screen has next appearance. 12](#_Toc257028699)

[Game entities 12](#_Toc257028700)

[Bonuses 12](#_Toc257028701)

[Bricks 13](#_Toc257028702)

[Paddle 13](#_Toc257028703)

[Walls 13](#_Toc257028704)

# Development goals

1. TDD learning and practicing. Basic variant – unit testes + stubs.
2. Lazy resource loading (e.g. loading resources only necessary for current level).
3. Study collision detection and building real-time game process rather that per-cycle.
4. Practice level change scheme, previously created. It also includes work with level tree.
5. (Excess???) Building complex menu, including in-game menu usage.
6. (Excess ???) Optimizing and using post-process effects (bloom, motion blur, bump mapping, water reflection).

# Jobs (task types)

1. *Programmer*. Unit testes, bug fixing, code writing, optimization, writing comments for (at least) own code.
2. *Project manager*. General project vision, development direction and current tasks.
3. *Team lead*. General architecture design, core programming, build assembling, tasks and bugs assigning.
4. *Programmer of effects and animation*. All complex animation, animation controllers, particle system and post-processing.
5. *GUI programmer*. All game menu.
6. *3D-designer + artist*. Creating/searching of 3D models, textures, fonts and other graphical content.
7. *Sounds and music*. Creating/searching of games sounds and music.
8. *Testers*. Test builds.
9. *Wiki editor*.

# Tools

1. *Programming language* – Java (<http://java.sun.com>)
2. *Unit testing* - JUnit (http://junit.sourceforge.net)
3. *Graphics engine* – jMonkeyEngine (<http://jmonkeyengine.com>)
4. *GUI* –GBUI (http://gbui.googlecode.com)
5. *IDE* – Eclipse (http://www.eclipse.org)
6. *3D editor* – Blender (<http://www.blender.org>)
7. *UML tool* – Visual Paradigm (http://www.visual-paradigm.com/product/vpuml/)

# Game overview

“Stones of History” is arcade, variant of breakout. Main feature is historical appearance. All game levels are grouped in epochs. **Epoch** roughly corresponds to the main epochs of our civilization or particular cultural epoch (prehistoric epoch, ancient Egypt, Greece, Rome etc). All epochs are sorted in chronological order.

Each level within epoch corresponds to some significant event. For each epoch and for each level short history information is displayed. So game can be used also for educational purposes.

Several epochs can be opened for playing at the same time (e.g. ancient Greece and Rome). Next epochs are opened only after completing all previous epochs. That is, if Greece epoch and epoch of Persian Empire are opened, the Epoch of Rome can’t be opened until they both became finished.

During game player acts as prominent (and not) historical persons. For example typical level intro is “You are Spartacus, the leader of slave rebellion. You were going to cross to Sicily, but you suffer a failure, and now Roman legions cut your army from the rest of Italy. The only way to stay alive is break though the enemy defense…”

The game field is a set of objects (bricks, stones, legionnaires) to be destroyed to go to the next level. Different types of bricks have different strength. After destroying a brick some bonuses can occur (e.g. plus one life, increase paddle with, super-ball etc). Bonuses can be both helpful and harmful.

A player wins when reach last level. Every time when ball intersects with bottom wall it takes one life. If there is no life player lose the game.

## Prototypes

* **Breaks of** **Egypt/Atlantis/Camelot**) **-** historical appearance, bonuses. (<http://games.skunkstudios.com/games/bricksofatlantis/>, <http://games.skunkstudios.com/games/bricksofcamelot/>, <http://games.skunkstudios.com/games/bricksofegypt/>).







* **Ballistik** – 3D graphics. (http://games.skunkstudios.com/games/ballistik/)

****

* **Magic Ball 3**–levels with some story elements, using bricks with complex shape (with some restrictions). (<http://games.skunkstudios.com/games/magicball3/>) 

****

## System requirements

Game is croosplatform. There is separate build for each platform (Windows, Linux, Mac OS X). Only 32-bit systems are supported.

60 FPS are expected on next configuration (it not minimal or recommended, just configuration of game developer)

|  |  |
| --- | --- |
| Parameter | Платформа розробника |
| Operating System | Windows XP, Windows 7, Linux (Kubuntu 8.10). |
| Processor | Core 2 Duo T5870 2,00GHz (one core) |
| RAM | 2Gb (actual 100-200Mb) |
| Free HDD space | 10-20 Mb |
| Graphical card | NVIDIA GeForce 8400M GS |
| Sound card | OpenAL compatible |
| Controls | Mouse, keyboard |
| Other | Java JRE 1.5 |

# Functional description

## Game screens

Whole game is a sequence of game screens. Transitions between these screens are represented on below.



### Loading

Screen with black background that has progress indicator and percentage of loaded process located at the center of the screen.

### Logo



### Menu

New game (Continue)

Options

Music on/off

Sounds on/off

Back

Controls

<List of control keys>

Back

Records

Credits

Exit

Records table shows 10 best scores. It is shown in text form:

Nickname score lines level

Aaa 123 12 2

Bsd 34 2 0

--- --- --- ---

### Credits

The list of words which is moved from bottom to top. It has format:

Job:

List of developers

Job:

List of developers

### Level selection



Here inaccessible items are marked with gray color.

### Game

### Game screen has next appearance.



## Game entities

### Bonuses

Bonuses can be collected and combined (e.i. can be several active bonuses at the same time). Bonuses can be persistent (e.g. additional life) or temporary (almost all other, e.g. width increase, super ball etc), which are work only during certain time (typically 10-20 seconds). Bonus is created after destroying a brick, which holds it, and falls down with constant speed. To active bonus player needs pick up it with paddle.

Bonus flavors:

* Additional life.
* Increase paddle width.
* Decrease paddle width.
* Increase ball speed.
* Decrease ball speed.
* Paddle with gun (paddle can shoot bullets, that act like balls, but move only forward).
* Sticky paddle (balls stick to paddle when contacts with it; it comes unstuck after user presses button)
* Bottom wall (new temporary wall is created under behind the paddle)
* Ball splitting (for each ball its clone is created, new direction of original ball is +45 degrees and direction of clone is -45 degrees from direction of original ball)

### Bricks

Bricks (or stones) – are main game elements. Every brick has its own 3D model and strength (number of stokes to destroy it). With each brick can be associated some bonus.

There are 3 types of bricks:

* ***Default.*** When hit by ball strength of brick is decreased by one and ball is repulsed under physics laws for light beams. When strength of bricks reaches zero, it is destroyed.
* ***Persistent.***Can’t be destroyed.
* ***Glass.***Has the same behavior as default brick, but ball is not repulsed but flies thought the brick. Has strength 1.

### Paddle

Paddle is the only user-controlled game element. It has flat front side. When hit by ball, new ball direction is calculated from the formula arccos((x-0.5l)/0.5l), where l – total length of paddle.

### Walls

When ball hits them its move direction is calculated under physics laws for light beams.