**Design document**

**Kazak and Alien, short interactive story**

**By Oleksandr Bondarev**

Kazak and Alien is a short interactive story. Application will start with title screen. Then instructions screen will be shown. After that, there will be meteor mini-game. Next screen will show more instructions. Next a picture of Kazak and nature will appear with a small UFO in left top corner. User will have to control it and land on hill on the right. If landed on a hill – an alien will move towards Kazak. After that another instructions menu will appear and user will have to solve small logic game. After that game ends. There are also 2 secret endings. You can see more details in initial proposal (last page of this document).

**Design**

**Main**

Contain global variables. Setup screen and call World class. Display World. Handle key and mouse events and send them to World class.

Properties

* Global colors
* Global coordinates
* Variable for World and Minim

Methods

setup()

* Initialize World class
* Setup screen size and background.

draw() - display World class.

keyPressed() - Send keyPressed action to World class

keyReleased() - Send keyReleased action to World class

mousePressed() - Send mousePressed action to World class

.

**World**

Contain information about current act number (information about acts can be seen in Initial Proposal in the end of the document). Depending on current act - call different classes, display them and send mouse and key events to appropriate classes

Properties

* current Act
* variables for the following classes: Instructions, Background, Alien, Foreground, Landing, Sound, Ufo, LogicGame, Collect

Methods

reset() - reset world to initial state for starting new world

display() – display classes from current act

handleKeyPressed() – handle keyPressed() from main class and send to classes depending on current act.

handleKeyReleased() - handle keyRelesed() from main class and send to classes depending on current act.

handleMousePressed() - handle mousePressed() from main class and send to classes depending on current act.

nextAct() – switch current act to next act.

**Collect**

Minigame where you need to catch or avoid meteors

Properties

* variables for game status
* Variables for Meteor, World, Sound and Ufo classes

Methods

reset() - reset game status to initial point

display() – display all data that is needed for this game

transparentBackground() - makes visible "tails" for moving objects

endAct() - analyze game status and check if game is over

displayEnergy() - show energy bar in top left corner of the screen

intersect() - analyze position of UFO and all meteors. If UFO intersects with one of the meteors -> meteor changes position and UFO's energy is updated

fallMeteors() - launch meteors one by one every "startPeriod" milliseconds update meteors position

startMeteors() - launch next meteor, until all are launched; update time period; update UFO's energy

**Background**

Draw sky, background hills and river

Properties

* main colors

Methods

display() – draw a sky and nature.

drawSky() – self explaining.

drawNature() – draw hills and river.

drawTree(float x, float y) - draws a tree in given coordinates

**Alien**

Draw an alien of different size, depending on act

Properties

* main colors
* time variables
* alien coordinates and velocity
* alien state
* Variable to store Sound and World classes

Methods

reset() – reset alien state.

display() - draws and moves small alien

displayTall() - draws and moves tall alien

displayCloser() - draws and moves tall alien, when he comes close

smallAlien(float aPosX, float aPosY) - draws a small alien in given coordinates

moveAlienToCoordinate(int trgtX, int trgtY) - moves alien to given coordinates changes alien status to next when arrived

moveToCoordinate(float cur, float trgt) - makes 1 step towards given coordinates. Step is taken according to aliens' velocity

nextState() - Change state of the alien to a next state

setX (float x), setY (float y) - set new coordinate

tallAlien(float aPosX, float aPosY) - draws tall alien in given coordinates

**AlienState** – enum that is used to show alien state

**Foreground**

Show foreground image with Kazak and big Hill

Properties

* variable for Kazak class

Methods

display() – display front hill and Kazak and, after some point, an Alien

drawFrontHill() – self explaining

**Instructions**

Handle instructions’ screens between mini-games

Properties

* Main coordinates
* Difficulty buttons' coordinates
* String variable to store instruction’s data
* Variables to store PFont, PImage, LogicGame, Sound and World classes

Methods

displayStart() - act#1 - displays start menu

spaceInfo() - act#2 - displays short information

landing() - act#4 - displays start menu

displayGuessInstructions() - act#5 - displays instructions about guess game

end() - act#8 - game ends here

secretPilot() - act#9 - secret ending #1 (great pilot)

secretShame() - act#10 - secret ending #2 (shame)

showTextWindow() - shows text window to make text more readable

showText(float x, float y, float tWidth, float tHeight, String text) - show given text in given coordinates, of given size, with one preset style

continueBtn(float x, float y) - draw continue button in given coordinates

noBtn(float x, float y) - draw "No!" (no help) button in given coordinates

handleMouseEvents() - handle mouse events in this act

act5MouseEvents() - handle mouse events in act#5

inButton(float x, float y) – return true, if button is in given coordinates and mouse is over, else - false

**Kozak -** Draw Kazak in red suite.

Properties – main colors

Methods

display() – display Kozak

**Landing**

Highlight all landing areas. Correct place is highlighted in blue

Properties

* main colors
* Boolean variables for class status

Methods

display() – display landing field if “L” button was pressed

handleKeyPressed() - Handle “L” key (pressed)

hide() – set status to switched off

opposite() – change status to opposite

**Ufo**

Draw Ufo. In act #3 and #5 it can be controlled by arrow keys on keyboard and mouse clicks

Properties

* Boolean variables for Ufo’s status
* Variables for class status
* variables for calculating Ufo’s movement
* variables for Alien, Landing, Sound, and World classes

Methods

reset() – reset UFO variables to starting point

display() – display UFO

moveUFO() – calculate next coordinates of UFO, based on current position and player controls

setLandingStatus() – provide Alien object with information about landing

handleKeyPressed() – handle keyPressed() from WorldModel class

handleKeyReleased() - handle keyRelesed() from WorldModel class

handleMousePressed() - handle mousePressed() from WorldModel class

**LogicGame**

Play small logic game, using a mouse

Properties

* variables for game logic
* variables for UI
* variable for World and Sound classes

Methods

reset(World world, Instructions inf) - reset game

play() – play logic game

makeGuess() – control pressed buttons and update game memory with chosen answer

displayUpBtn(float x, float y), displayDownBtn(float x, float y) – displays up or down button in given coordinates

guessBox(int x, int y, String val) – display square in given coordinates with chosen value

displayConfirmBtn(float x, float y) - displays a confirm button in given coordinates

changeFillColor(float x, float y), changeCircleFillColor(float x, float y) – change buttons color based on given coordinates and mouse coordinates

btnUp(),btnDown() - displays up or down button

givenBtnIsSelected(float x, float y), confirmIsSelected(float x, float y) – return true if buttons in given coordinated intersect with mouse coordinates

generateRandomCorrectVariant() – self explaining

guessToZero() – set all values of guess array to 0

setHistory() – save guess to array with history

handleMousePressed() - handle mousePressed() from WorldModel class

setGuess() – update guess value based on mouse coordiantes

confirmGuess() – check current guess after confirm button was pressed

processGuess() – process guess values

processWinScenario(), processLooseScenario() – finish game with one of the scenarios

noFirstTime() – is needed to prevent winning from the first try

increaseDifficulty(), decreaseDifficulty() – change game difficulty

**Meteor**

Show meteors in act#3 mini-game. 1 object for each meteor

Properties

* class status

Methods

fall() - update meteors position

getQuality(), getX(), getY(), getSize() – return data about meteor

start() - make meteor moving

collect() - hide meteor and make it explode

**Sound**

Work with sound

Properties

* sound file names
* path and type of sound files
* variable to store AudioPlayer class

Methods

play(String name) - play selected sound. Some sounds will be played only after others finish other sounds will play, as soon as needed

stop() - stop playing current sound

switchSong(String name) - switch current song to given one

**Initial proposal**

**Concept**: This is going to be a short interaction story-game about an Alien and Kozak. It will be divided in several acts.

**Act#1**, “Start screen”: with application name

**Act#2,** “Introduction”: User will see some text with short story, objective and control keys that are needed to fly UFO. He will also see information about meteors: which meteors should be caught and which – avoided.

**Act#3, “**Catch meteors”: catch and avoid different meteors

**Act#4**, “Landing Instructions”: some more instructions about landing UFO in next act

**Act#5**, “Landing UFO”: Some nature is shown. A Kazak in red suite is drawn in the left bottom corner of the screen. UFO is drawn in the left top corner of the screen and starts to fall down. User will control UFO. User has to land it according to an objective (far right hill). He or she will use controls described in act#1. If objective is complete small alien appears from the UFO and starts moving towards Kazak. Until he comes near.

**Act#6**, “Logic game Instructions”: User will see instructions for small game and its rules.

**Act#7**, “Guess ingredients”: Small logic game where user has limited tries to guess which ingredients are used to generate energy for UFO. Every game will need random ingredients. There will be hints: how many ingredients were guessed, but user will have to guess which are correct. It is impossible to guess in first turn.

**Act#8**, “Finish”: Game results. User can start game from the logical game or close it.

**Act#9,** “Great pilot ending”: secret ending. If you succeed in act#3

**Act#10,** “Shame ending”: secret ending. If player refuses to help in act#6

**Interaction**: There will be arrow keys and mouse clicks for UFO controlling. Clicking mouse on screen will also control everything else. There will be different buttons, depending on act number.

**Objectives**: Help an alien to get gas and fly away in space. Learn how to use arrays, classes and complex logic of switching between acts.

**References**

Dahno, V. (1987). How kozaks met aliens. Retrieved March 28, 2016, from <http://multik.usemind.org/147-kak-kazaki-inoplanetyan-vstrechali>

Dahno, V. (1987). How kozaks played Olympics. Retrieved March 28, 2016, from <http://mults.info/mults/?id=125>

Devaney, L. (2015, November 18). *New Minecraft tutorial teaches kids coding.* Retrieved March 24, 2016, from <http://www.eschoolnews.com/2015/11/18/minecraft-tutorial-059/>

Hyperspace jump. (n.d.). Retrieved March 28, 2016, from <http://www.shutterstock.com/video/clip-8264476?language=en>

Processing 1.0 - Processing Discourse - Rain effect. (n.d.). Retrieved March 30, 2016, from <https://processing.org/discourse/beta/num_1275997615.html>

Project video part 1: <http://screencast-o-matic.com/watch/cDevfx1UFz>

Project video part 2: <http://screencast-o-matic.com/watch/cDevfp1UFK>