Finalized class design

Two UML diagram come with this documentation. One with the full diagram and one with just the core classes (Interfaces and Physics package removed).

Interfaces used only to separate the Model View and Controller part of the application.

- Event Handler: from View to Controller
- IEditorView | IPlayView: from Controller to View
- IGameModel | IGameTimer | IGizmo | IGizmoFactory: from Controller and View to Model
- Observer: from Model to View

Changes

- View representation changed
- Classes get renamed
- Controllers grouped by segments of UI instead of 1 controller for 1 view
- Timing functionality received his own class
- Enums became inner classes

Class rearrangments: EditController -> ToolModeEventHandler | TopToolbarEventHandler | GizmoSelectorEventHandler GameController, PlayController -> GameBarEventHandler | IngameKeyEventHandler | PauseMenuEventHandler SaveHandler -> GameLoader | GameSaver

MainController DELETED IngameMenuView DELETED MainView DELETED FlipperDirection -> Moved insed Flipper class.

Old Name	New name
AddGizmo	AddGizmoStrategy
ConnectGizmo	ConnectGizmoStrategy
DeleteGizmo	DeleteGizmoStrategy
RotateGizmo	RotateGizmoStrategy
IngameMenuController	PauseMenuEventHandler
FlipperDirection	Flipper.Orientation
PhysicsBodyFactory	GizmoFactory
PhysicsBody	Gizmo
PhysicsWorld	GameModel
ShapeFactory	GizmoDrawer

View classes

Gizmoball

The main application class. Responsible to launch up the application.

PlayView and EditView

This two class meant to represent the UI of the editor mode(EditView) and the play mode(PlayView). Also these classes responsible to execute any changes regarding the UI such as opening dialogues, drawing boards or switching the UI layout from play to editor the other way around.

GizmoDrawer

GizmoDrawer is a class which is responsible for drawing out the gizmos to a canvas.

IEditorView | IPlayView

Provides an interface for our UI classes so other parts of the applications can request their services without knowing the actual implementation of the UI or what UI library is in use.

Observer

This interface is used in our Views to implement the observer part of the Observer design pattern. For UI refreshing when the state of the game is changed.

Controller classes

GameLoader

This class is responsible for opening and reading the .gizmo files.

GameSaver

This class is responsible for opening and saving the game to a file.

GameBarEventHandler | IngameKeyEventHandler | PauseMenuEventHandler

This controller classes meant to handle the events generated by the play mode UI. Each controller handles different section's events. GameBarEventHandler -> small ingame toolbar IngameKeyEventHandler -> keyboard events PauseMenuEventHandler -> the events from the small pause menu.

ToolModeEventHandler | TopToolbarEventHandler | GizmoSelectorEventHandler

This Controller classes handlers the events coming from each section of the editor UI. ToolModeEventHandler -> the left side tool selector bar TopToolbarEventHandler -> the top big toolbar GizmoSelectorEventHandler -> the right side add gizmo selector

AddGizmoStrategy | ConnectGizmoStrategy | RemoveGizmoStrategy | RotateGizmoStrategy | MoveGizmoStraegy

These controller classes meant to represent the different event handling modes for the editor UI's canvas clicks and drags.

EventHandler

This interface used to hide the actual controller implementation from the view when the controllers are called by the view.

Model classes

Gizmo

Superclass for all gizmos. Representing those traits which are common in every Gizmo.

Absorber | Ball | CircleGizmo | Flipper | Square | Triangle

These classes are actual implementation of the basic gizmos. Each of them represents one gizmo. These classes are extending the Gizmo Class.

Walls

Special gizmo class represents the 4 wall around the playing area.

GameMode

This class responsible to hold all the gizmos which are currently represented in the playing area and handle their physical behaviour based on time.

CollusionDetails

Helper class for collision detection and handling. Stores information about one collusion such as time until the collision, the actual gizmo which going collide with or target and the new speed after collision.

GameTimer

This class is responsible to provide a steady heartbeat for the game. To make sure the GameModel is updated on each beat.

GizmoFactory

GizmoFactory is for creating objects for gizmos classes. Mostly creating gizmos based on the result of incoming user events from editor UI (trough a controller), or file reading from GameLoader.

Dot

Multiple instances of this class used together to represent the physical body of the gizmos for drawing purposes.

IMovable

Provides the necessary functions for those gizmos which meant to be able to move in the physics loop.

ITriggerable

This interface marks those gizmos which are meant to be triggered.

ITrigger

Interface for those gizmos which are meant to be able to trigger other gizmos and provides methods for connecting them.

IGameModel | IGameTimer | IGizmo | IGizmoFactory

This interfaces used to hide the actual implementation from another part of the application (Controller, View).