Paradisa

Cattubene
Yuchen Jiang, Yuchen Zhou,
Zhimin Sun, Rundong Tang

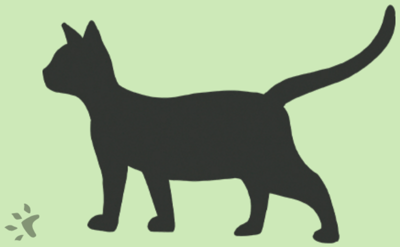
What have we learnt?

1. **Technical**
   (a) Coding structure and coding quality
   (b) Functional programing (elm)
   (c) The usage of Web Front End: Html and Svg
   (d) LaTeX
2. **Teamwork**
   (a) Branches of coding:
   The usage of Github and atomic commits
   (b) Distribution of codes
   Distribute coding works by layers and do program management.
3. **Design**
   (a) Game Design:
   User experience and mechanisms oriented design.
   (b) Booklet Design:
   Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

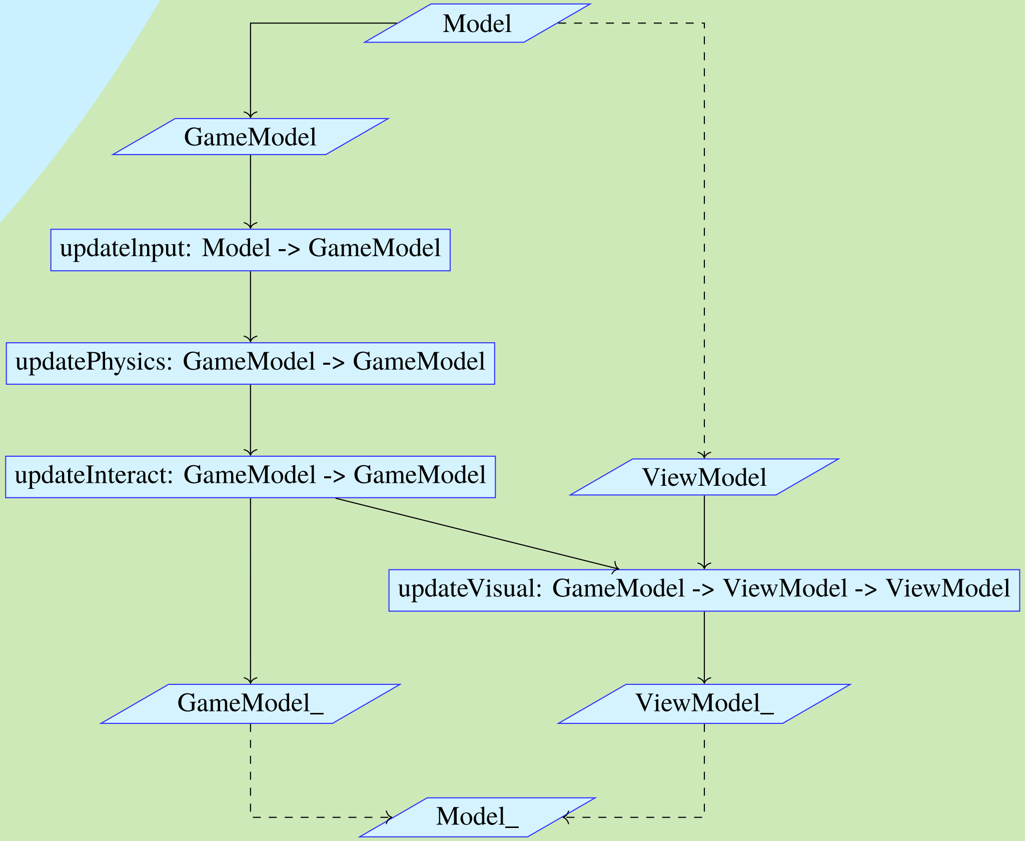


Achievements

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**



Model

GameModel

updateInput: Model -> GameModel

updatePhysics: GameModel -> GameModel

updateInteract: GameModel -> GameModel

ViewModel

updateVisual: GameModel -> ViewModel -> ViewModel

GameModel_

ViewModel_

Model_

3. **Clear Debugger for Development**

   Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.

Paradisa

Cattubene
Yuchen Jiang, Yuchen Zhou,
Zhimin Sun, Rundong Tang

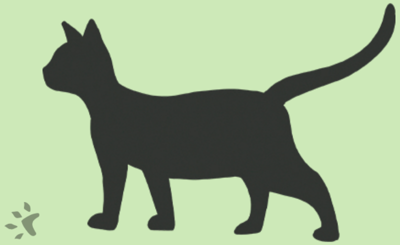
What have we learnt?

1. **Technical**
   - (a) Coding structure and coding quality
   - (b) Functional programing (elm)
   - (c) The usage of Web Front End: Html and Svg
   - (d) LaTeX
2. **Teamwork**
   - (a) Branches of coding:
     The usage of Github and atomic commits
   - (b) Distribution of codes
     Distribute coding works by layers and do program management.
3. **Design**
   - (a) Game Design:
     User experience and mechanisms oriented design.
   - (b) Booklet Design:
     Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

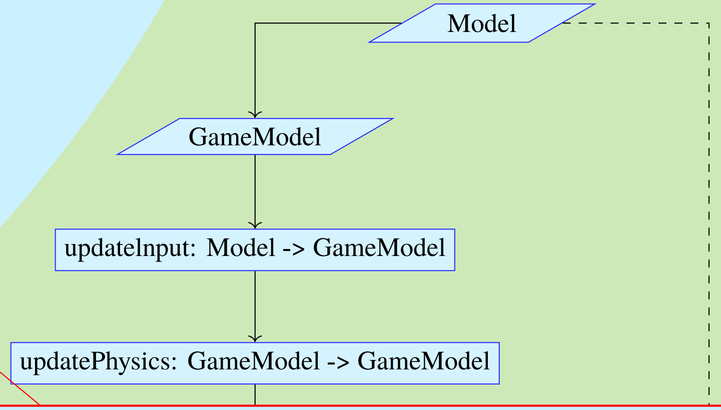


Achievements

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**


Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
odel

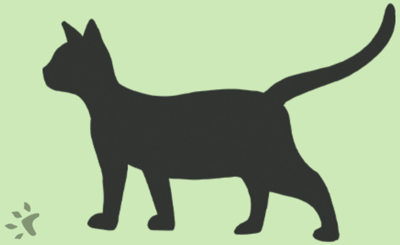
insight to the code.

What have we learnt?

1. **Technical**
   - (a) Coding structure and coding quality
   - (b) Functional programing (elm)
   - (c) The usage of Web Front End: Html and Svg
   - (d) LaTeX

Paradisa

Cattubene
Yuchen Jiang, Yuchen Zhou,
Zhimin Sun, Rundong Tang



What have we learnt?

1. **Technical**
   - (a) Coding structure and coding quality
   - (b) Functional programing (elm)
   - (c) The usage of Web Front End: Html and Svg
   - (d) LaTeX
2. **Teamwork**
   - (a) Branches of coding:
     The usage of Github and atomic commits
   - (b) Distribution of codes
     Distribute coding works by layers and do program management.
3. **Design**
   - (a) Game Design:
     User experience and mechanisms oriented design.
   - (b) Booklet Design:
     Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

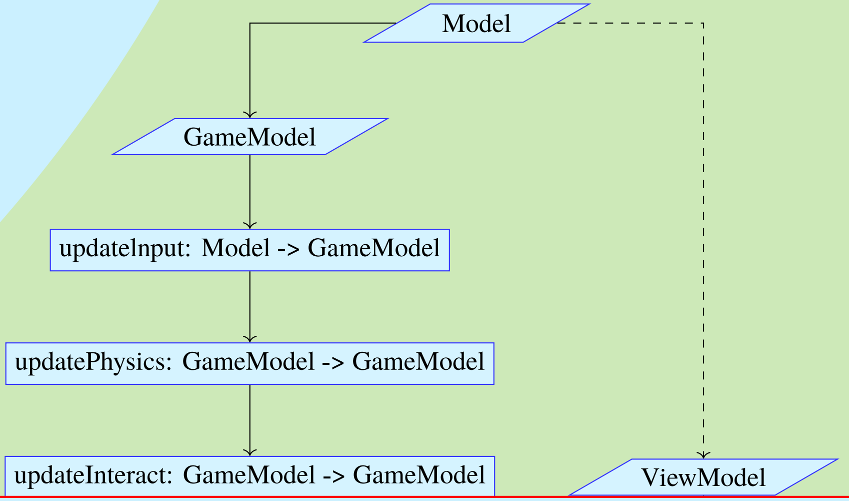
Improvement Throughout Course

1. New Way of Thinking in Functional Programm
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in comp
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

Achievements

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.
2. **Creative Application of Classic MVC Model**

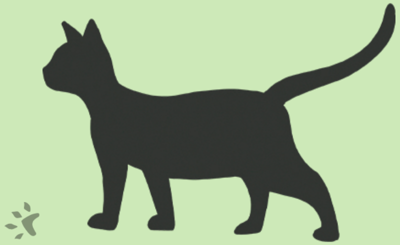

Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
updateInteract: GameModel -> GameModel
ViewModel
odel

insight to the code.

2. **Teamwork**
   - (a) Branches of coding:
     The usage of Github and atomic commits
   - (b) Distribution of codes
     Distribute coding works by layers and do program management.

Paradisa

Cattubene
Yuchen Jiang, Yuchen Zhou,
Zhimin Sun, Rundong Tang



What have we learnt?

1. **Technical**
   1. (a) Coding structure and coding quality
   2. (b) Functional programing (elm)
   3. (c) The usage of Web Front End: Html and Svg
   4. (d) LaTeX
2. **Teamwork**
   1. (a) Branches of coding:
      The usage of Github and atomic commits
   2. (b) Distribution of codes
      Distribute coding works by layers and do program management.
3. **Design**
   1. (a) Game Design:
      User experience and mechanisms oriented design.
   2. (b) Booklet Design:
      Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

Improvement Throughout Course

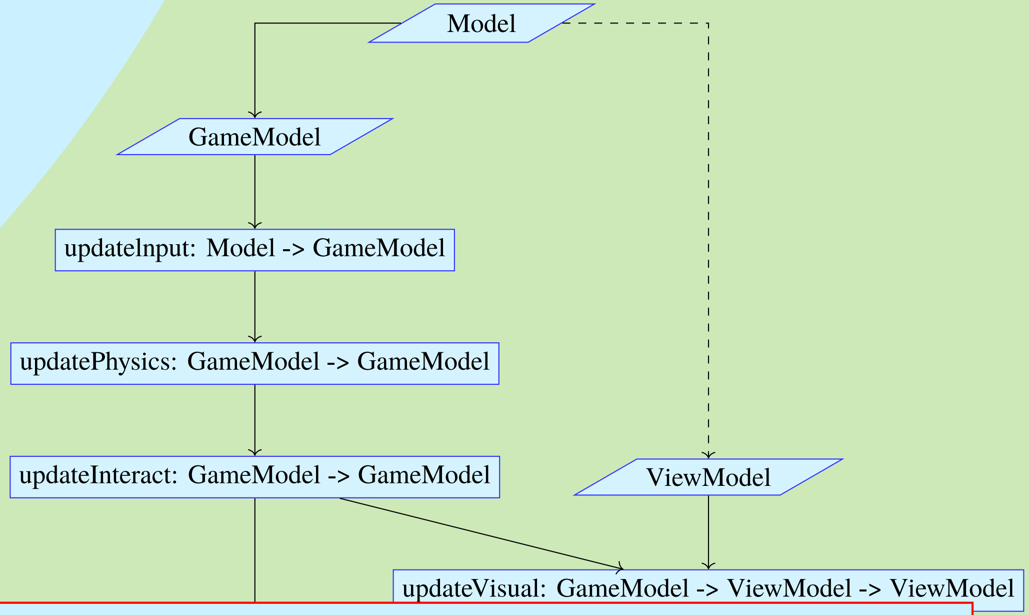
1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companie
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

Achievements

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**



Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
updateInteract: GameModel -> GameModel
ViewModel
updateVisual: GameModel -> ViewModel -> ViewModel

insight to the code.

3. **Design**
   (a) Game Design:
   User experience and mechanisms oriented design.
   (b) Booklet Design:
   Delicate instructional booklet design using Photoshop.

Paradise

4. **Working as a Developer in a Company**

In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

What have we learnt?

1. **Technical**
   - (a) Coding structure and coding quality
   - (b) Functional programing (elm)
   - (c) The usage of Web Front End: Html and Svg
   - (d) LaTeX
2. **Teamwork**
   - (a) Branches of coding:
     The usage of Github and atomic commits
   - (b) Distribution of codes
     Distribute coding works by layers and do program management.
3. **Design**
   - (a) Game Design:
     User experience and mechanisms oriented design.
   - (b) Booklet Design:
     Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

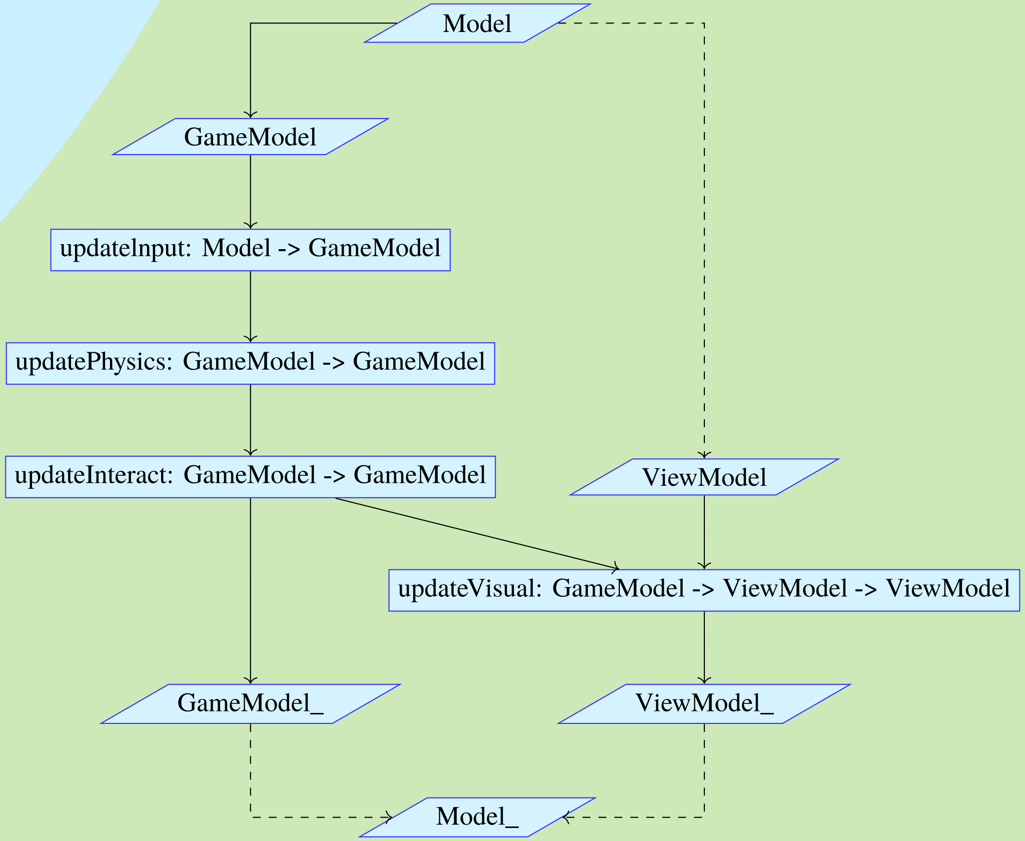
   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**


Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
updateInteract: GameModel -> GameModel
ViewModel
updateVisual: GameModel -> ViewModel -> ViewModel
GameModel_
ViewModel_
Model_

3. **Clear Debugger for Development**

   Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.

Paradise

What have we learnt?

1. **Technical**
   (a) Coding structure and coding quality
   (b) Functional programing (elm)
   (c) The usage of Web Front End: Html and Svg
   (d) LaTeX
2. **Teamwork**
   (a) Branches of coding:
   The usage of Github and atomic commits
   (b) Distribution of codes
   Distribute coding works by layers and do program management.
3. **Design**
   (a) Game Design:
   User experience and mechanisms oriented design.
   (b) Booklet Design:
   Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

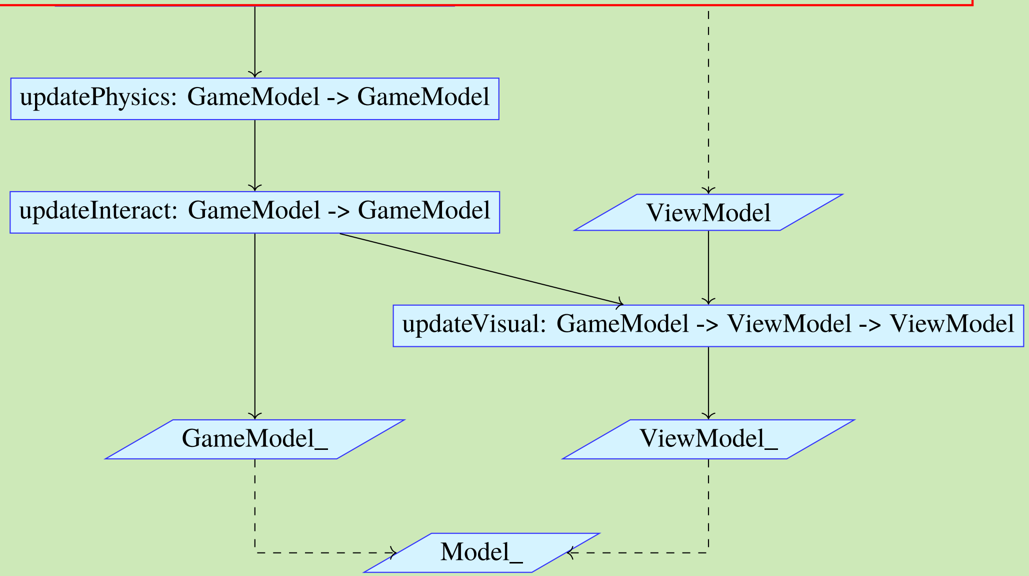
   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette



updatePhysics: GameModel -> GameModel

updateInteract: GameModel -> GameModel

ViewModel

updateVisual: GameModel -> ViewModel -> ViewModel

GameModel_

ViewModel_

Model_

3. **Clear Debugger for Development**

   Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.

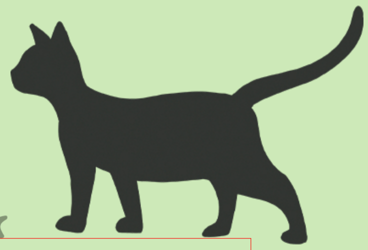
Paradisa

Cattubene
Yuchen Jiang, Yuchen Zhou,
Zhimin Sun, Rundong Tang

What have we learnt?

1. **Technical**
   (a) Coding structure and coding quality
   (b) Functional programing (elm)
   (c) The usage of Web Front End: Html and Svg
   (d) LaTeX
2. **Teamwork**
   (a) Branches of coding:
   The usage of Github and atomic commits
   (b) Distribution of codes
   Distribute coding works by layers and do program management.
3. **Design**
   (a) Game Design:
   User experience and mechanisms oriented design.
   (b) Booklet Design:
   Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

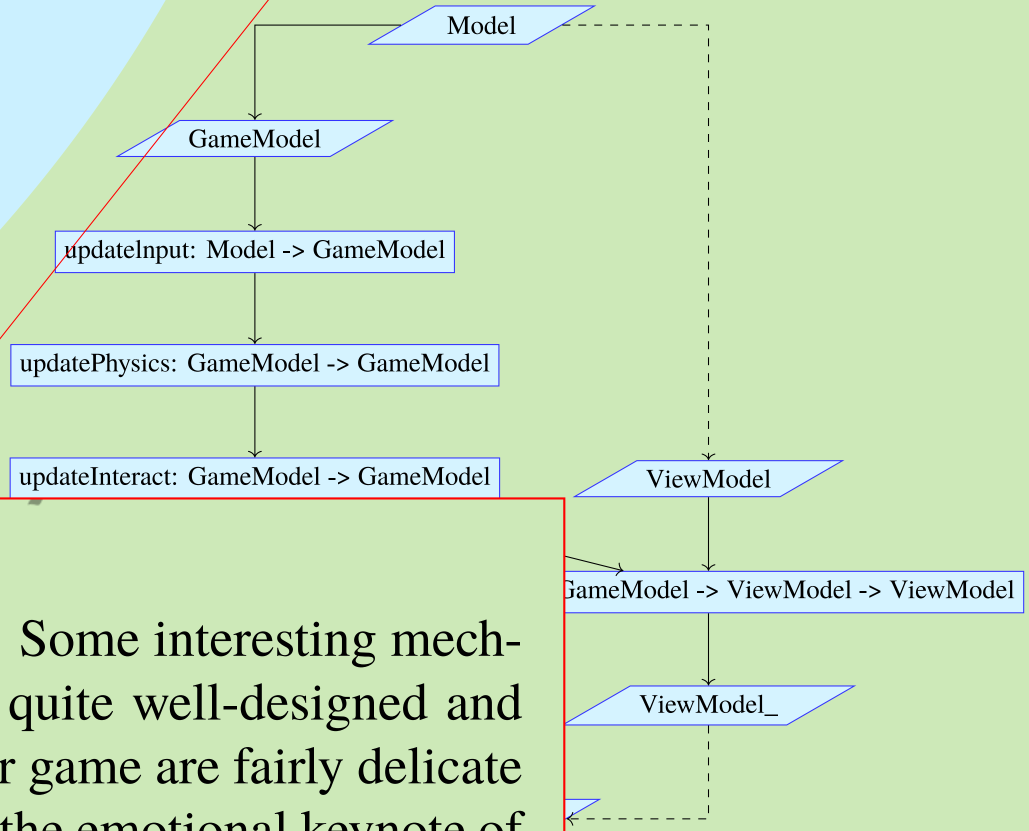


Achievements

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**


Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
updateInteract: GameModel -> GameModel
ViewModel
GameModel -> ViewModel -> ViewModel
ViewModel_

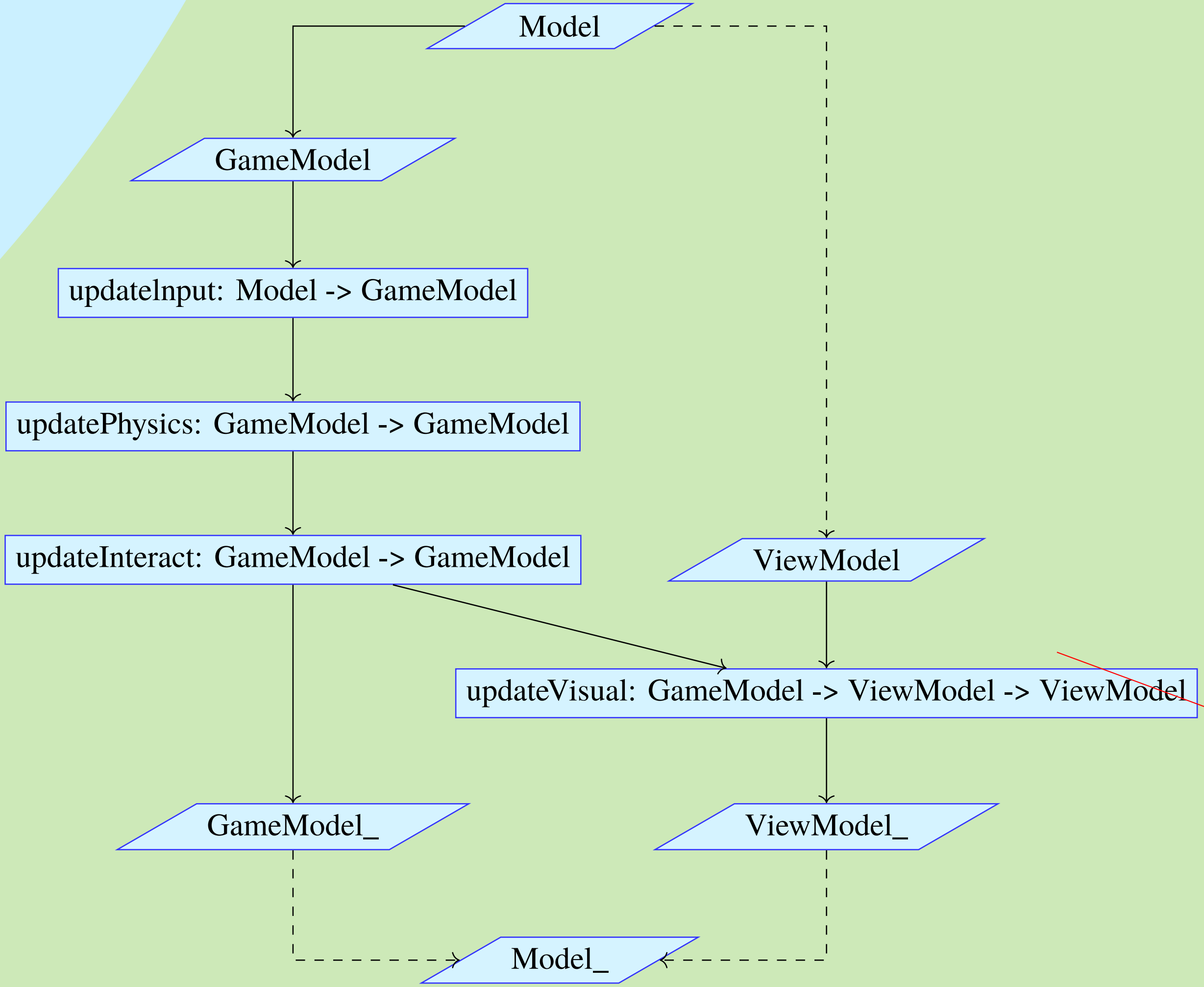
1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

ing speed and a deeper insight to the code.

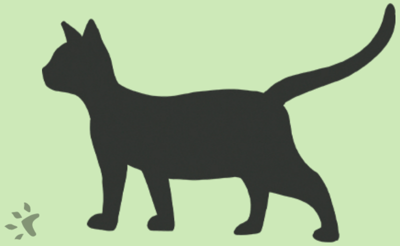
Paradise

2. **Creative Application of Classic MVC Model**


Model
GameModel
updateInput: Model -> GameModel
updatePhysics: GameModel -> GameModel
updateInteract: GameModel -> GameModel
ViewModel
updateVisual: GameModel -> ViewModel -> ViewModel
GameModel_
ViewModel_
Model_

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

ubene
, Yuchen Zhou,
Rundong Tang

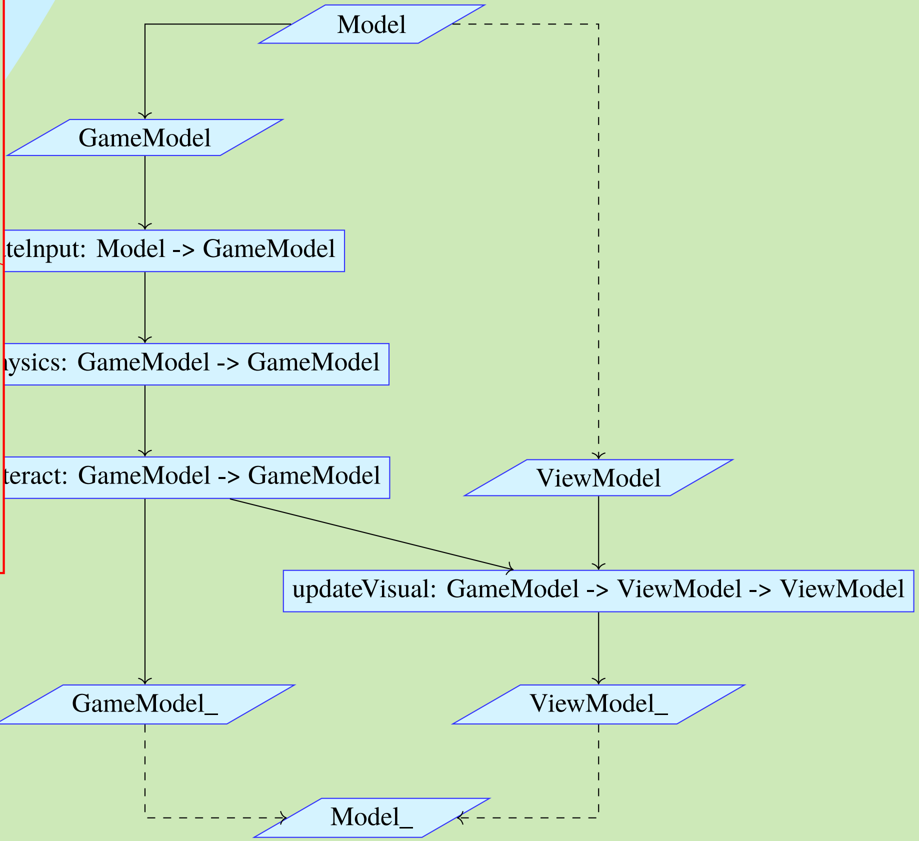


ents

ete Game with Delicate Artworks

te 2D RPG game with original elements. Some interesting mech- e applied to our game levels, which are quite well-designed and) attract players. Also, the artworks in our game are fairly delicate / a unified style, which is consistent with the emotional keynote of story.

Application of Classic MVC Model


Model
GameModel
teInput: Model -> GameModel
ysics: GameModel -> GameModel
teract: GameModel -> GameModel
ViewModel
updateVisual: GameModel -> ViewModel -> ViewModel
GameModel_
ViewModel_
Model_

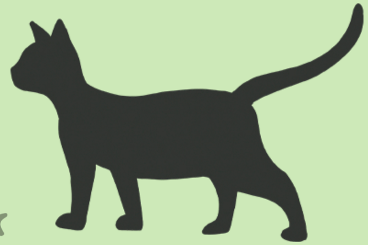
3. **Clear Debugger for Development**

   Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.

Paradise

3. **Clear Debugger for Development**

Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.



1. **Technical**
   (a) Coding structure and coding quality
   (b) Functional programing (elm)
   (c) The usage of Web Front End: Html and Svg
   (d) $\LaTeX$
2. **Teamwork**
   (a) Branches of coding:
   The usage of Github and atomic commits
   (b) Distribution of codes
   Distribute coding works by layers and do program management.
3. **Design**
   (a) Game Design:
   User experience and mechanisms oriented design.
   (b) Booklet Design:
   Delicate instructional booklet design using Photoshop.
4. **Working as a Developer in a Company**

   In Project 2, we are required to use Redmine to act as developers in a company. which helps to reduce risks and adapt us to the possible environment where we work in the future.

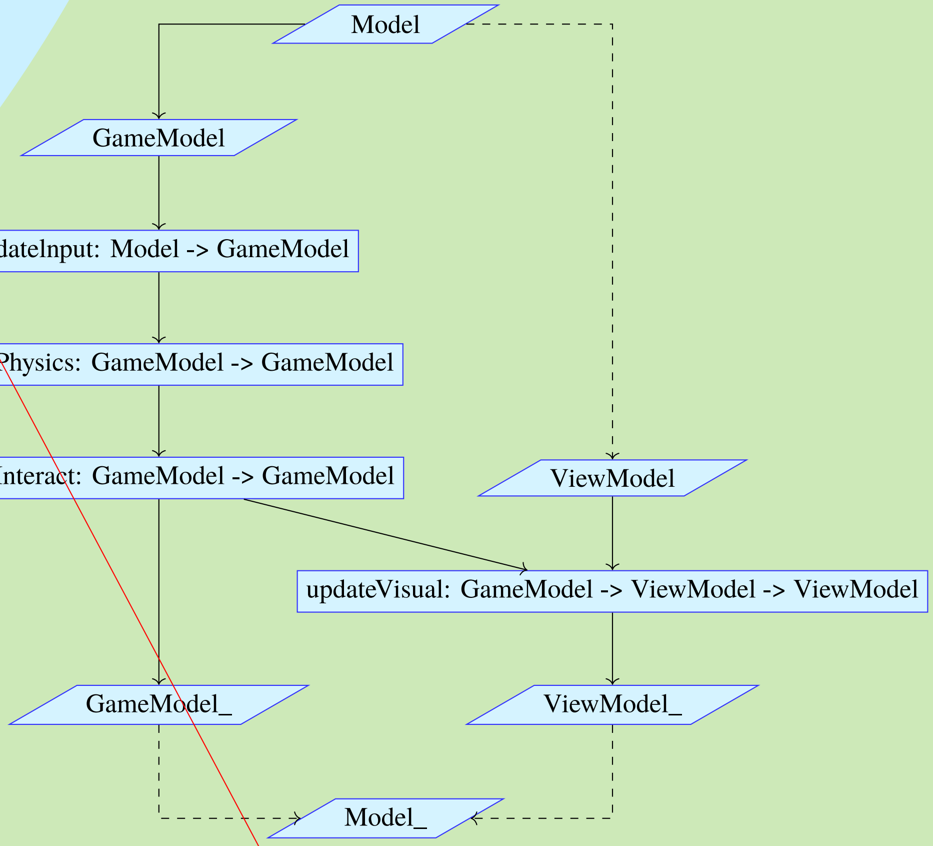
Improvement Throughout Course

1. New Way of Thinking in Functional Programming
   - Data driven
   - Pipeline structures
   - Static Types means safer.
2. The importance of standard procedure in companies
3. Unified Artistic Style
4. The importance of Copyright and Netiquette

1. **A Complete Game with Delicate Artworks**

   A complete 2D RPG game with original elements. Some interesting mechanisms are applied to our game levels, which are quite well-designed and are sure to attract players. Also, the artworks in our game are fairly delicate and follow a unified style, which is consistent with the emotional keynote of our game story.

2. **Creative Application of Classic MVC Model**



Model

GameModel

updateInput: Model -> GameModel

updatePhysics: GameModel -> GameModel

updateInteract: GameModel -> GameModel

ViewModel

updateVisual: GameModel -> ViewModel -> ViewModel

GameModel_

ViewModel_

Model_

3. **Clear Debugger for Development**

   Demonstration of data flow enables a faster debugging speed and a deeper insight to the code.