Programming Lab

Programming Design Ideas

Program Design Tools

- There are many tools to help you think about and design a program
    - these will be taught in a class later
    - but we can start to use them in a simple way now

- CRC Cards

- Use Cases

CRC Cards

- These are used to help think about what Objects or Classes are needed by the program

- CRC
  - C : class   - name of the class
  - R : responsibility   - what is the responsibility
  - C : collaboration   - who does it work with

CRC Cards

- These are used to help think about what Objects or Classes are needed by the program

- CRC
  - C : class   - name of the class
  - R : responsibility - what is the responsibility
  - C : collaboration - who does it work with

Deck
_____

- responsible for cards that will be drawn for one player

_____

Player, CardFactory, GameManger

CRC Cards

- These are used to help think about what Objects or Classes are needed by the program

- CRC
  - C : class   - name of the class
  - R : responsibility - what is the responsibility
  - C : collaboration - who does it work with

Deck

- responsible for cards that will be drawn for one player

Player, CardFactory, GameManger

Notice that the responsibility does not tell about the functions or data at this point

CRC Cards

- These are used to help think about what Objects or Classes are needed by the program

- CRC
    - C : class   - name of the class
    - R : responsibility - what is the responsibility
    - C : collaboration   - who does it work with

Deck
_____

- responsible for cards that will be drawn for one player

_____

Player, CardFactory, GameManger

Collaborations will be be found in the next step

Use Cases

- Actions that the system/program can take
  - make a list of all the actions that can happen in the game/program

Use Cases

- Actions that the system/program can take
    - make a list of all the actions that can happen in the game/program

for example: end the game, play card

Use Cases

- Actions that the system/program can take
  - make a list of all the actions that can happen in the game/program
  - for each action, what are the classes that will be used to do this action?

Use Cases

- Actions that the system/program can take
    - make a list of all the actions that can happen in the game/program
    - for each action, what are the classes that will be used to do this action?
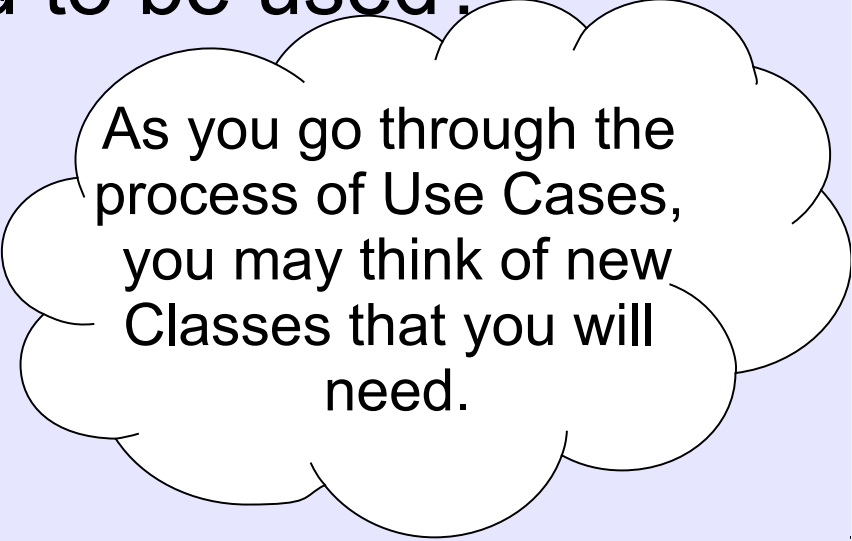
for example: to "end the game" what classes from our CRC Card will need to be used?

Use Cases

- Actions that the system/program can take
  - make a list of all the actions that can happen in the game/program
  - for each action, what are the classes that will be used to do this action?

for example: to "end the game" what classes from our CRC Card will need to be used?

As you go through the process of Use Cases, you may think of new Classes that you will need.

Use Cases

- Actions that the system/program can take
  - make a list of all the actions that can happen in the game/program
  - for each action, what are the classes that will be used to do this action?

for example: to "end the game" what classes from our CRC Card will need to be used?

you may want to add new collaborations to your CRC cards as you notice them while doing the Use Cases

Working on the Classes

- Once you have finished the CRC Cards and Use Cases, you can start to think about what data and methods the class will need.

Working on the Classes

- Once you have finished the CRC Cards and Use Cases, you can start to think about what data and methods the class will need.

- This information can come from it's responsibilities and what other objects will want from this class (collaborations)
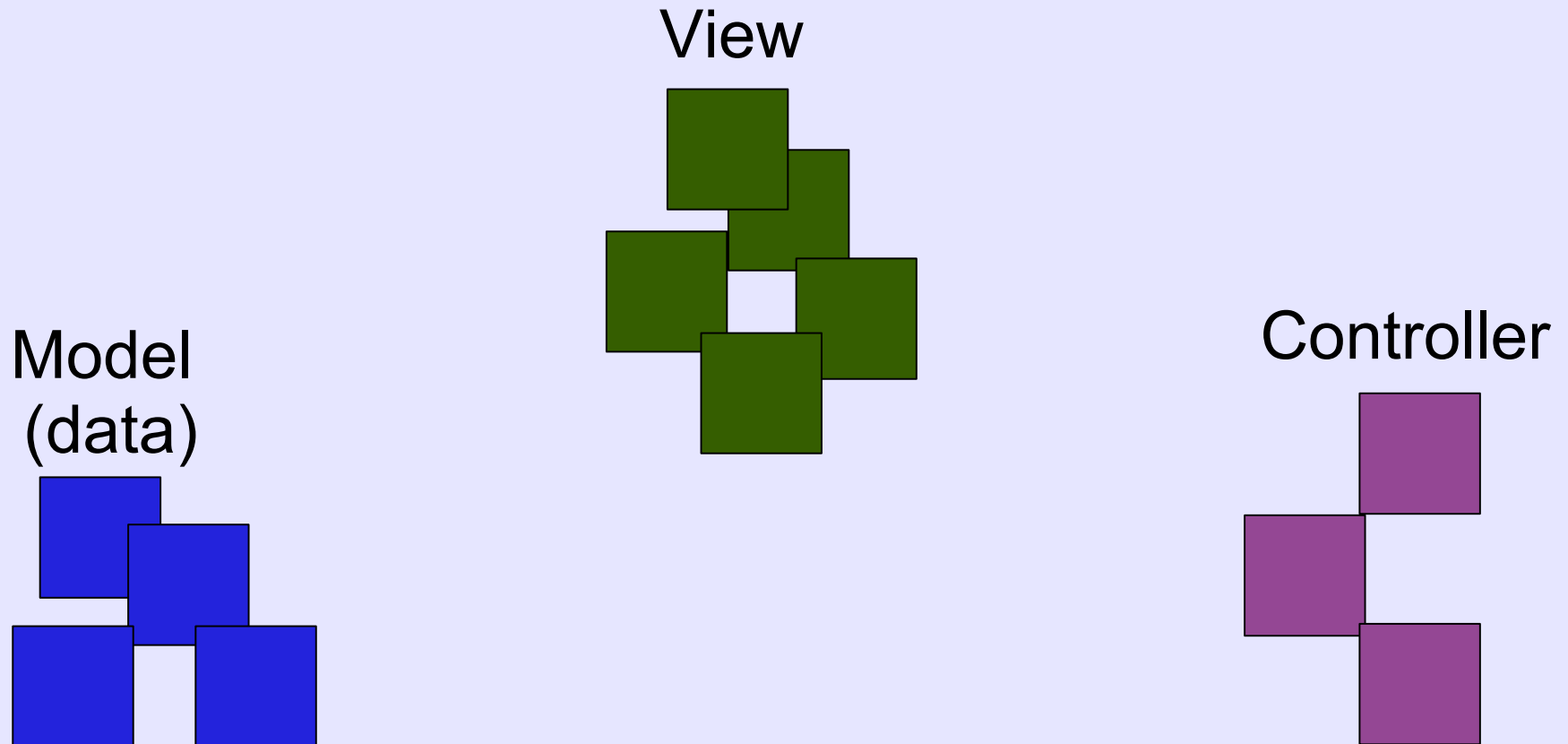
Thinking about the GUI

- A common way to think of the program and the GUI is with the MVC model

- M   Model: the classes that manage game information (data)

- V   View: the GUI that shows the game

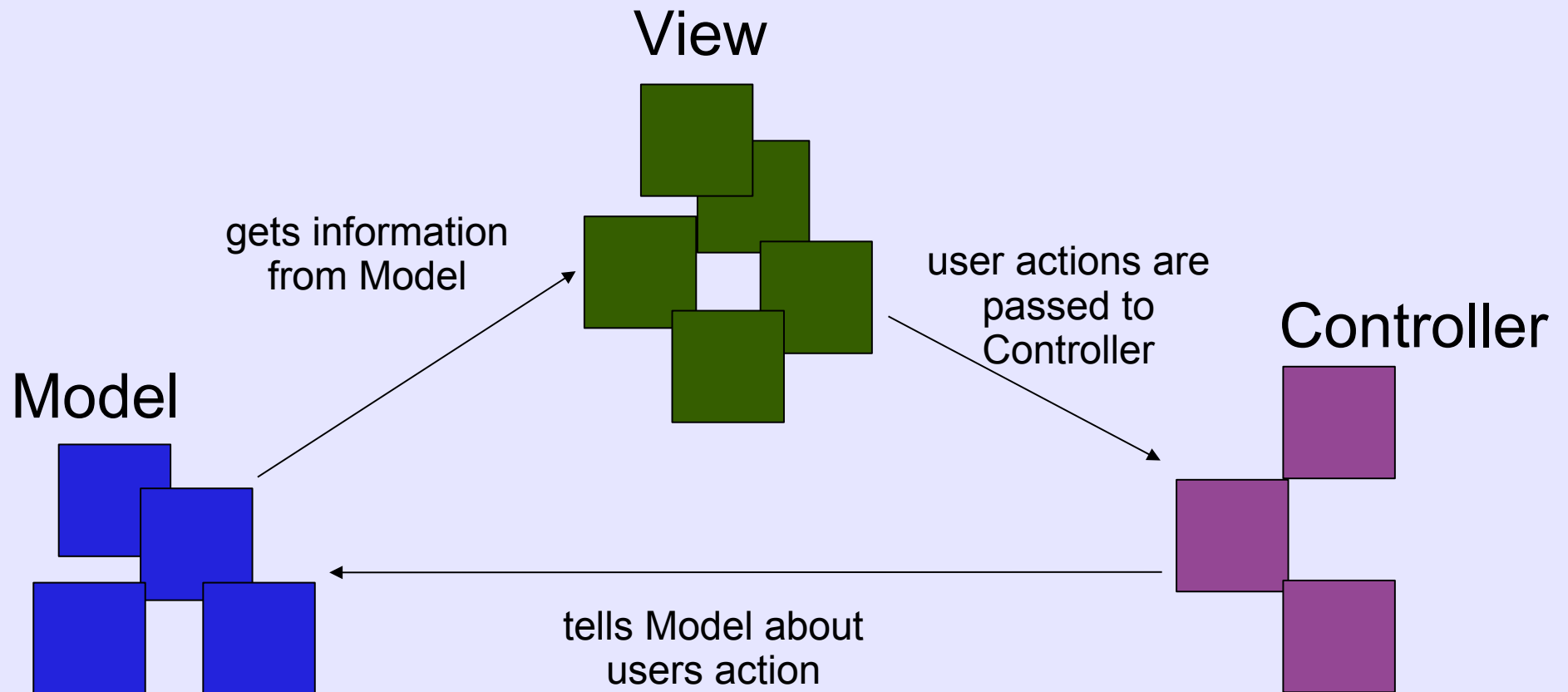- C   Controller: the event handlers that react to user input

Thinking about the GUI

- A common way to think of the program and the GUI is with the MVC model

View

Controller

Model
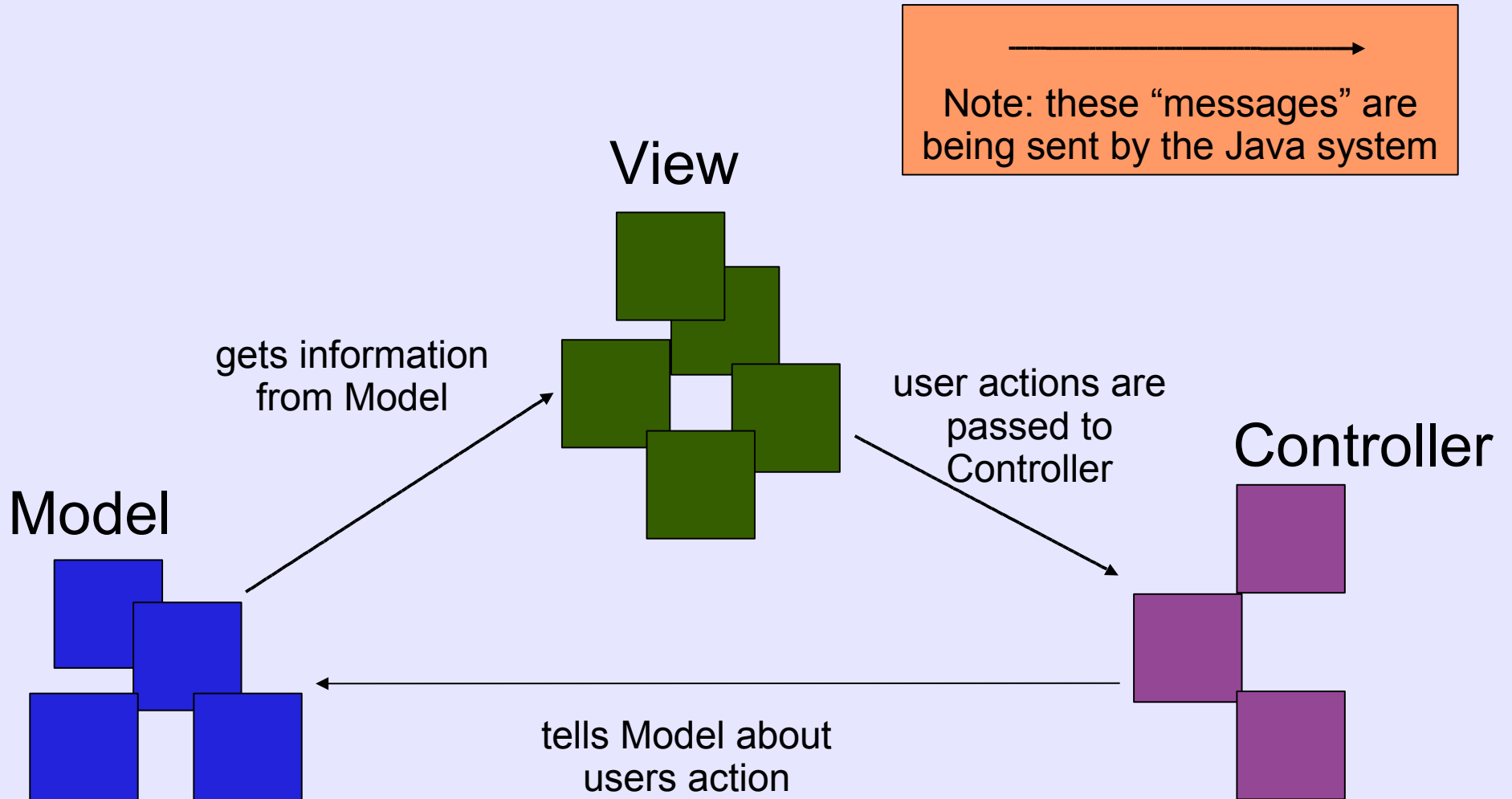(data)

Thinking about the GUI

- A common way to think of the program and the GUI is with the MVC model

View

gets information
from Model

user actions are
passed to
Controller

Controller

Model

tells Model about
users action

Thinking about the GUI

View

Note: these "messages" are being sent by the Java system

gets information from Model

user actions are passed to Controller

Controller

Model

tells Model about users action

Thinking about the GUI

View

Note: these "messages" are being sent by the Java system

Model

~New~
Binding
Properties

~Changed~
Event
handlers

Controller

tells Model about
users action