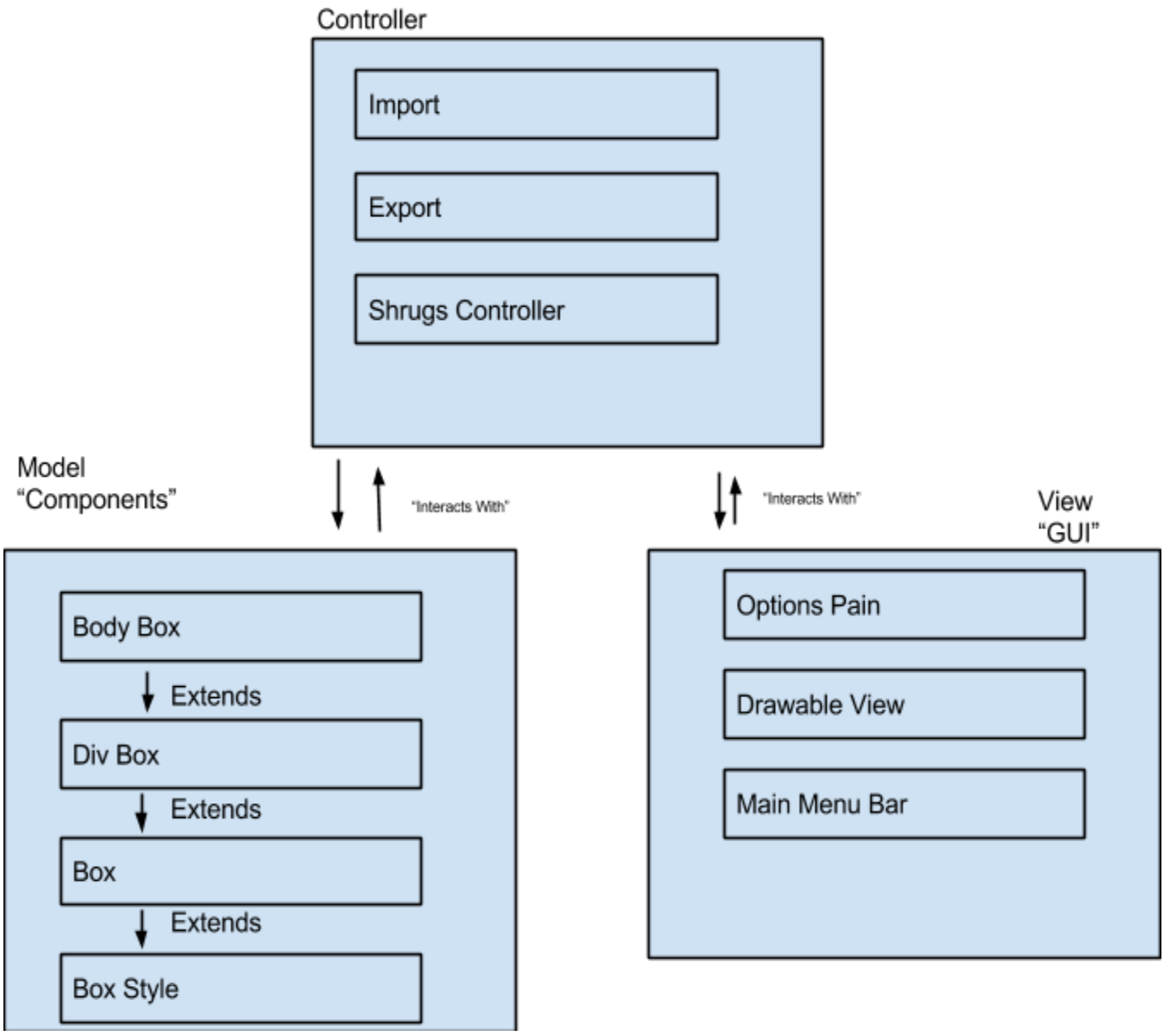
Shrugs

Shrugs Helps Regular Users Generate Sites

Sprint 2: Incrememtal/Regression Testing

Classification of Components / Hierarchy



Outline of Components; Input & Output of components; and Dependent Components

Base Components:

- ***Model (Components)***
  - **Independent Components:**
    - **Box** (Superclass for all Box Subclasses) - Describes a box in terms of its two corners and its parent.
      - Input: Two corners, a Style object, a parent Box
      - Output: A valid Box object
    - **DivBox** (Extends the Box Object) - Includes a list of child boxes
      - Input: Two corners, a Style object, a parent Box, any number of child Boxes
      - Output: A valid DivBox object
    - **BodyBox** (Extends the DivBox Object) - The top-level box
      - Input: Two corners, a Style object
      - Output: A valid BodyBox object
    - **BoxStyle** - A set of parameters that correspond to their CSS equivalents including font type, color, font size
      - Input: A valid Box object
      - Output: A BoxStyle object containing all of the saved CSS parameters
- ***View (GUI)***
  - **Independent Components:**
    - ***ToolBar*** - A graphical toolbar that will display various options that can be set for a given selected *Box* object or any of its subclasses.
      - Input: Takes in a *Box* component or any of its subclasses.
      - Output: A list of *Box* attributes which can be edited via user mouse/keyboard input from the *ToolBar*.
      - Dependent Components: This component calls the *ShrugsController* component to handle the modification of the *Box* object model data. The *View (GUI)* and *ShrugsController* components both can call this component.
    - ***DrawableView*** - A graphical canvas pane which takes in user mouse input to interact with the ShrugsController to modify the *Box* object model data.
      - Input: Takes in user mouse clicks, mouse drags, and mouse releases and a flag to decide which *Box* component to create.
      - Output: A *Box* component of subclass type based on the flag passed in.
      - Dependent Components: This component calls the

ShrugsController component to handle the creation of the *Box* object model data. The *View (GUI)* component calls this component.

- ***MainMenuBar*** - A graphical menu bar which takes in user mouse click input. The MainMenuBar contains MenuItems which interact with each of the Controller's independent components. Namely the *Import, Export,* and *ShrugsController* components.
  - Input: Takes in user mouse clicks to decide what components to call.
  - Output: A call to a component respective to the menu item clicked by the user.
  - Dependent Components: This component can call either the *ShrugsController, Export,* or *Import* components of the *Controller* base component. *The View (GUI)* component is what calls this component.

- ***Controller***
  - **Independent Components:**
    - ***Import*** - This component handles the process of importing a Shrugs project file into the Shrugs editor. This component interfaces and interacts directly with the *Model* to create/update *Box, BodyBox, and DivBox* objects.
      - Input: Takes in a user-specified file location of a Shrugs project file.
      - Output: A list of *Model (Components)* to populate the *DrawableView* with.
      - Dependent Components: This component directly calls the *Model* component to handle the creation of *Box, DivBox, and BodyBox* components. The *MainMenuBar* component calls this component.
    - ***Export*** - This component handles the process of exporting a Shrugs project file to HTML and CSS. This component interfaces and interacts directly with the *Model* to convert *Box, BodyBox, and DivBox* objects into corresponding HTML and CSS based on the *Model's* data.
      - Input: Takes in a user-specified file location of a Shrugs project file and a list of all *Model (Components) to* export.
      - Output: Packaged HTML and CSS files.
      - Dependent Components: This component directly calls the *Model* component to handle the conversion of *Box, DivBox, and BodyBox* components to HTML and CSS. The *MainMenuBar* component calls this component.
    - ***ShrugsController*** - This component is the the main interaction with the *Model*. It handles the creation/updating/deletion actions to be performed on the *Model (Components)* base components as well as all of it's sub-independent components.
      - Input: Takes in *Model (Components), Box, DivBox, and BodyBox*

components.

- Output: Creation/updating/deletion of input *Model* data.
- Dependent Components: This component directly calls the *Model* component to handle the creation/updating/deletion of *Box, DivBox, and BodyBox* components. The *View (GUI)* calls this component.

We chose to, once again, test our application using a bottom-up incremental testing approach. In this approach, we started testing the very lowest individual components of the software using drivers, and then worked upwards, integrating base components incrementally going up the software hierarchy. In order to do this, we had to write drivers and use user GUI input to provide the test input for the modules being testing (Until we reached the top of the hierarchy).

MODULE: BoxStyle
INCREMENTAL TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
	getColor() needs to return a Color object instead of a String.		Use regular expression to parse String and save the values into a Color object.

REGRESSION TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
	GUI is not able to render colors from Strings		Check if return value from getColor is a valid Color object.

MODULE: Components

INCREMENTAL TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	If the Box is drawn with the one of its sides existing as another (i.e. they share the same x or y coordinate) the box will not be drawn.	1	Set a distinction so that if lines do exist on the same plane, then the object is allowed.
2	If the Box is drawn with an endpoint that exists outside of the project base diagram, it will create an object that will have part of its body inaccessible.	2	Set a drag boundary for the project background so that the cursor cannot be set out of bound.
3	Sometimes when creating a child for a Box, the child will not be assigned to the direct parent Box, but instead to a grandparent Box.	1	Have the box set the closest relative parent to the child. this can be done with a basic loop of all existing boxes.
4	The parent box object is not recognized by the child.	1	simply not implemented. can be implemented via once child is set, access the child object box and set it as the parent.

REGRESSION TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	The Object will not be exported when the parent of the object is not set.	1	Set the parents of the children to allow for export to work.
2	When the tool bar is moved, the objects will not recognize the resize of the project grid and will not.	2	Have the boxes resize based on percentages of the total number of pixels the height and width are.
3	If a box object of height or width of 0 exist. Then a different Box object can’t be made within the vicinity, causing a “dead area” of	1	set it so box objects can’t have a width or height of 0.

	the project.		

MODULE: DrawableView

INCREMENTAL TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	Boxes are instantiated incorrectly.	1	Change (x1,y2,w,h) to (x1,y1,x2,y2)
2	Null color, produced by canceling color chooser dialogue, can disrupt box drawing.	2	Propagate the null color exception to the drawable view.
3	A box drawn with a width or height of zero can be created. This is undesirable behavior.	2	Check the width and height of box while drawing and set variable “make” to false if either width or height is 0.

REGRESSION TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	After correcting box instantiation, boxes	1	Draw boxes using *width()*/*height()*

	are not drawn correctly.		instead of *endX/endY*.
2	Default to previous color chosen if color chooser is canceled.	2	Capture the null color exception and default to the previously chosen color, or Color.WHITE if no previous color was chosen.
3	Ensure that boxes with width or height of zero cannot be created.		Check if make == false. If that is the case, then do not create the box as it does not satisfy all the constraints. (Including width/height constraints)

MODULE: ToolBar

INCREMENTAL TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	The attributes (JComponents) within the toolbar should dynamically adjust and resize to fill the toolbar.	2	Use a layout manager inside the toolbar to ensure the components are laid out properly.
2	The components inside the toolbar should be dynamically populated/updated based on input from the toolbar combobox.	2	Create new JComponents and add them to the JToolBar component to allow dynamic updating of components. Handle this on the combobox

			action event.

REGRESSION TESTING

DEFECT #	DESCRIPTION	SEVERITY	HOW TO CORRECT
1	Fix the toolbar scale issue.	2	Use dimensions of the main menu and a layout manager to ensure the toolbar continues to scale on main window resizing.
2	Fix component mis-population bug.	2	Override the toolbar's paint method to ensure that the components are not only created, but redrawn as well.