SE 3XA3: Module Interface Specification
Save The Date

Karuka Khurana (khurak1)
Utsharga Rozario (rozariou)
Samarth Kumar (kumars38)
Dhruv Cheemakurti (cheemakd)

March 18, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
March 14	1.0	Initial React Doc By Utsharga
March 16	1.0	Updated React Doc By Utsharga Updated Python Doc By Samarth
March 18	1.1	Updated React Doc By Utsharga Updated Python Doc By Samarth

1 editablePage Module

1.1 Template Module extends React.Component

editablePage

1.2 Uses

editableBlock, uid, caretHelpers

1.3 Description

This module represents the page. It details the suite of functions that are able to be executed on a page: editing, adding and deleting. It is therefore a controller module.

1.4 Syntax

1.4.1 Exported Constants

initialBlock : EditableBlock

1.4.2 Exported Types

ReactDOM

1.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
updatePageHandler	EditableBlock		
addBlockHandler	EditableBlock		
deleteBlockHandler	EditableBlock		
render		ReactDOM	

1.5 Semantics

1.5.1 State Variables

props : HTML attribute
updatedBlock : EditableBlock
currentBlock : EditableBlock

1.5.2 State Invariant

None

1.5.3 Assumptions

None.

1.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = initialBlock$
- output: *out* := self
- exception: none

updatePageHandler(updatedBlock):

- transition: This function updates an editable block if the user makes any changes to the block.
- exception: none

addBlockHandler(currentBlock):

- transition: This function adds addition blocks if the user enters a new block.
- exception: none

deleteBlockHandler(currentBlock):

- transition: This fucntion deletes the block the user choose to delete.

- exception: none

render():

- output: This function renders the output of the editable page.
- exception: none

2 editableBlock Module

2.1 Template Module extends React.Component

editableBlock

2.2 Uses

selectMenu, caretHelpers

2.3 Description

This module represents a block in the page. It details the suite of functions that are able to be executed on a block: editing, adding and deleting. It is therefore a controller module.

2.4 Syntax

2.4.1 Exported Constants

CMD_KEY : Character

2.4.2 Exported Types

ReactDOM

2.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
componentDidMount			
componentDidUpdate	EditableTable.state		
onChangeHandler	keystroke		
onKeyDownHandler	keystroke		
onKeyUpHandler	keystroke		
openSelectMenuHandler			
closeSelectMenuHandler			
tagSelectionHandler	String		
render		ReactDOM	

2.5 Semantics

2.5.1 State Variables

props : HTML attribute
prevState : EditableTable.state

2.5.2 State Invariant

e : keystroke

2.5.3 Assumptions

None.

2.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = null$
- output: $out :=$ self
- exception: none

componentDidMount():

- transition: This function sets the current state to the object state.
- exception: none

componentDidUpdate(prevState):

- transition: This function updates the page components if the user has either changed the html content or the tag.
- exception: none

onChangeHandler(e):

- transition: This function handles any changes to the HTML content of an EditableBlock.
- exception: none

onKeyDownHandler(e):

- transition: This function handles any changes to the block from the user using key strokes.
- exception: none

onKeyUpHandler(e):

- transition: This function handles the SelectMenu on KeyUp input from the user.
- exception: none

openSelectMenuHandler():

- transition: This function operates after opening the SelectMenu, it then attaches a click listener to the dom.
- output: This function closes the menu after the next click - regardless of outside or inside menu.
- exception: none

closeSelectMenuHandler():

- transition: This function changes state of the SelectMenu.
- output: This function closes the SelectMenu.
- exception: none

tagSelectionHandler(tag):

- transition: This function assigns the tag to the EditableBlock.
- exception: none

render():

- output: This function renders the output of the EditableBlock and SelectMenu
- exception: none

3 selectMenu Module

3.1 Template Module extends React.Component

selectMenu

3.2 Uses

None

3.3 Description

This module represents a menu in the page. It details the suite of functions that are able to be executed on a menu: selecting. It is therefore a controller module.

3.4 Syntax

3.4.1 Exported Constants

allowedTags : Array of String
MENU_HEIGHT : Number

3.4.2 Exported Types

ReactDOM

3.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
componentDidMount			
componentDidUpdate	selectMenu.state		
componentWillUnmount			
keyDownHandler	keystroke		
render		ReactDOM	

3.5 Semantics

3.5.1 State Variables

props : HTML attribute
updatedBlock : EditableBlock
currentBlock : EditableBlock

3.5.2 State Invariant

e : keystroke

3.5.3 Assumptions

None.

3.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = null$
- output: $out :=$ self
- exception: none

componentDidMount():

- transition: This function attaches a key listener to add any given key to the command
- exception: none

componentDidUpdate(prevState):

- transition: This function checks whenever the command changes and looks for matching tags in the allowed list.
- exception: none

componentWillMount():

- transition: This function unmounts the key listener.

- exception: none

onKeyDownHandler(e):

- transition: This function hands the user input through key strokes.
- exception: none

render():

- output: This function renders the output of the select menu option
- exception: none

4 button Module

4.1 Template Module extends React.Component

button

4.2 Uses

EditableBlock, EditableTable, loadPDF, scrapePDFUser, image, SelectMenu

4.3 Description

This module represents the button on the side of every block. It details the suite of functions that are able to be executed on the block. It is therefore a controller module.

4.4 Syntax

4.4.1 Exported Constants

None.

4.4.2 Exported Types

ReactDOM

4.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
OptionHander	keystroke		
render		ReactDOM	

4.5 Semantics

4.5.1 State Variables

None.

4.5.2 State Invariant

None.

4.5.3 Assumptions

e : keystroke

4.5.4 Access Routine Semantics

constructor($props$):

- transition: $state = initialBlock$
- output: $out :=$ self
- exception: none

OptionHander(e):

- output: This function obtains the user input as one of the available options which it then initiates.
- exception: none

render():

- output: This function renders the button and the selectMenu on click.
- exception: none

5 editableTable Module

5.1 Template Module extends React.Component

editableTable

5.2 Uses

Table, loadData, utils, colors

5.3 Description

This module represents a table in the page. It details the suite of functions that are able to be executed on a menu: adding, editing and deleting. It is therefore a controller module.

5.4 Syntax

5.4.1 Exported Constants

None.

5.4.2 Exported Types

ReactDOM

5.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
reducer	Table.state, Table.action		
render		ReactDOM	

5.5 Semantics

5.5.1 State Variables

props : HTML attribute
state : Table.state
action : Table.action

5.5.2 State Invariant

None

5.5.3 Assumptions

None.

5.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = null$
- output: $out :=$ self
- exception: none

reducer(state, action):

- transition: This function is is used to reduce the actions performed on the EditableTable.
- exception: none

render():

- output: This function returns the rendering of the EditableTable.
- exception: none

6 Table Module

6.1 Template Module extends React.Component

Table

6.2 Uses

Cell, Header

6.3 Description

This module represents a table in the page. Its function is to only render the table itself. It is therefore a boundary module.

6.4 Syntax

6.4.1 Exported Constants

defaultColumn : Array of String

6.4.2 Exported Types

ReactDOM

6.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	2D Array, Array		
useMemo		Number	
useTable			
isTableResizing		Boolean	
render		ReactDOM	

6.5 Semantics

6.5.1 State Variables

data : 2D Array
columns : Array

6.5.2 State Invariant

None

6.5.3 Assumptions

None.

6.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = data, columns$
- output: $out :=$ self
- exception: none

useMemo():

- output: This function sets up all of the rows and columns.
- exception: none

useTable():

- transition: This function adjusts all of the sizing of the table.
- exception: none

isTableResizing():

- output: This function checks if the table needs resizing.
- exception: none

render():

- output: This function renders the outcome of the Table.
- exception: none

7 Header Module

7.1 Template Module extends React.Component

Header

7.2 Uses

utils, Text, colors

7.3 Description

This module represents a Header of a Table in the page. It details the suite of functions that are able to be executed on a menu: selecting. It is therefore a controller module.

7.4 Syntax

7.4.1 Exported Constants

buttons : Array of Strings
types : Array of Strings

7.4.2 Exported Types

ReactDOM

7.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
setExpanded	Boolean		
setHeader	String		
handleKeyDown	keystroke		
handleChange	keystroke		
handleBlur	keystroke		
render		ReactDOM	

7.5 Semantics

7.5.1 State Variables

props : HTML attribute
label : String

7.5.2 State Invariant

e : keystroke

7.5.3 Assumptions

None.

7.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = null$
- output: *out* := self
- exception: none

setExpanded():

- transition: This function sets up the Header to be expandable.
- exception: none

setHeader(label):

- transition: This function sets the Header to the label.
- exception: none

handleKeyDown():

- transition: This function handles the user input on key down strokes.
- exception: none

handleChange(e):

- transition: This function handles changes in the user input through key strokes.
- exception: none

handleBlur(e):

- transition: This function updates the Header.
- exception: none

render():

- output: This function renders the outcome of the Header.
- exception: none

8 Cell Module

8.1 Template Module extends React.Component

Cell

8.2 Uses

utils, colors

8.3 Description

This module represents a Cell in a Table in the page. It details the suite of functions that are able to be executed on a menu: selecting, editing and deleting. It is therefore a controller module.

8.4 Syntax

8.4.1 Exported Constants

None

8.4.2 Exported Types

ReactDOM

8.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
setValue	String		
dataDispatch	Cell		
handleOptionKeyDown	keystroke		
handleAddOption	keystroke		
handleOptionBlur	keystroke		
render		ReactDOM	

8.5 Semantics

8.5.1 State Variables

props : HTML attribute
value : string
dataCell : Cell

8.5.2 State Invariant

e : keystroke

8.5.3 Assumptions

None.

8.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = null$
- output: *out* := self
- exception: none

setValue(value):

- transition: This function set the value based on input.
- exception: none

dataDispatch(dataCell):

- transition: This function updates data using dataCell.
- exception: none

handleOptionKeyDown(e):

- transition: This function handles KeyDown options from user input.
- exception: none

handleAddOption(e):

- transition: This function handles additions rows added to the column.
- exception: none

handleOptionBlur(e):

- transition: This function updates data based on the key stroke.
- exception: none

render():

- output: This function renders the output of the Cell.
- exception: none

9 Relationship Module

9.1 Template Module extends React.Component

Relationship

9.2 Uses

None

9.3 Description

This module represents a Relationship in the Cell of a Table in the page. It function to only render the relationship. It is therefore a boundary module.

9.4 Syntax

9.4.1 Exported Constants

colors

9.4.2 Exported Types

ReactDOM

9.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
render		ReactDOM	

9.5 Semantics

9.5.1 State Variables

None.

9.5.2 State Invariant

None.

9.5.3 Assumptions

None.

9.5.4 Access Routine Semantics

render():

- output: *ReactDOM*
- exception: none

10 Text Module

10.1 Template Module extends React.Component

Text

10.2 Uses

None

10.3 Description

This module represents a Text in the Cell of a Table in the page. It function to only return a color as String. It is therefore a entity module.

10.4 Syntax

10.4.1 Exported Constants

None

10.4.2 Exported Types

ReactDOM

10.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
render		ReactDOM	

10.5 Semantics

10.5.1 State Variables

None.

10.5.2 State Invariant

None.

10.5.3 Assumptions

None.

10.5.4 Access Routine Semantics

render():

- output: *ReactDOM*
- exception: none

11 Colors Module

11.1 Template Module extends React.Component

Colors

11.2 Uses

None

11.3 Description

This module represents a colors in the Cell of a Table in the page. It function to only render the relationship. It is therefore a entity module.

11.4 Syntax

11.4.1 Exported Constants

None

11.4.2 Exported Types

ReactDOM

11.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
grey	Number	Number	

11.5 Semantics

11.5.1 State Variables

value : Number

11.5.2 State Invariant

None.

11.5.3 Assumptions

None.

11.5.4 Access Routine Semantics

grey(value):

- output: *out* = String
- exception: none

12 uid Module

12.1 Template Module

uid

12.2 Uses

None

12.3 Description

This module represents a uid generated for EditablePage and EditableBlock. It function to only output a string to other functions. It is therefore a entity module.

12.4 Syntax

12.4.1 Exported Constants

None

12.4.2 Exported Types

ReactDOM

12.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
uid		String	

12.5 Semantics

12.5.1 State Variables

None.

12.5.2 State Invariant

None.

12.5.3 Assumptions

None.

12.5.4 Access Routine Semantics

uid():

- output: *out* = String
- exception: none

13 caretHelpers Module

13.1 Template Module

caretHelpers

13.2 Uses

None

13.3 Description

This module gets and sets coordinates for EditableBlock. It is therefore a controller module.

13.4 Syntax

13.4.1 Exported Constants

None

13.4.2 Exported Types

ReactDOM

13.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
getCaretCoordinates		Number, Number	
setCaretToEnd	Object		

13.5 Semantics

13.5.1 State Variables

element : Object

13.5.2 State Invariant

None.

13.5.3 Assumptions

None.

13.5.4 Access Routine Semantics

getCaretCoordinates():

- output: *out* = Number, Number
- exception: none

setCaretToEnd(element):

- transition: This function sets the caret to the end of the existing components.
- exception: none

14 utils Module

14.1 Template Module

utils

14.2 Uses

None

14.3 Description

This module generates short IDs and random colours. It is therefore a entity module.

14.4 Syntax

14.4.1 Exported Constants

None

14.4.2 Exported Types

ReactDOM

14.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
shortId		String	
randomColor		String	

14.5 Semantics

14.5.1 State Variables

None.

14.5.2 State Invariant

None.

14.5.3 Assumptions

None.

14.5.4 Access Routine Semantics

shortId():

- output: This function creates a short unique ID from strings.
- exception: none

randomColor():

- output: This function generates random colours.
- exception: none

15 loadPDF Module

15.1 Template Module extends React.Component

loadPDF

15.2 Uses

None.

15.3 Syntax

15.3.1 Exported Constants

None.

15.3.2 Exported Types

ReactDOM

15.3.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
onDocumentLoadSuccess	Number		
changePage	Number		
previousPage			
nextPage			
render		ReactDOM	

15.4 Semantics

15.4.1 State Variables

props : HTML attribute
numPages : Number
offset : Number

15.4.2 State Invariant

None

15.4.3 Assumptions

None.

15.4.4 Access Routine Semantics

constructor(*props*):

- transition: $state = initialBlock$
- output: $out :=$ self
- exception: none

onDocumentLoadSuccess(numPages):

- transition: This function sets the number of pages and the initial page number to 1
- exception: none

changePage(offset):

- transition: This function calcuates the current page.
- exception: none

previousPage():

- transition: This function calculates the previous page by decrementing the current page.
- exception: none

nextPage():

- transition: This function calculates the next page by incrementing the current page.
- exception: none

render():

- output: This function renders the output of the PDF.
- exception: none

16 scrapePDFUser Module

16.1 Template Module extends React.Component

scrapePDFUser

16.2 Uses

loadData, EditableTable, PDFScraper

16.3 Description

This module represents the pop-up window when the user wants to scrape a document. It details the suite of functions that are able to be executed on the window: adding course name and selecting the PDF. It is therefore a controller module.

16.4 Syntax

16.4.1 Exported Constants

courseName : String
pageStart : Number
pageEnd : Number

16.4.2 Exported Types

ReactDOM

16.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
getCourseName		String	
getPageStart		Number	
getPageEnd		Number	
getFile			
scrape			
render		ReactDOM	

16.5 Semantics

16.5.1 State Variables

props : HTML attribute
numPages : Number
offset : Number

16.5.2 State Invariant

None

16.5.3 Assumptions

None.

16.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = initialBlock$
- output: *out* := self
- exception: none

getCourseName():

- output: This function obtains the Course Name from the User
- exception: none

getPageStart():

- output: This function obtains the starting page to scrap from the User
- exception: none

getPageEnd():

- output: This function obtains the end page to scrape from the User
- exception: none

getFile():

- transition: This function obtains the file to scrape from the User.
- exception: none

scrape(fileName):

- transition: This function uses the user inputs from getCourseName, getPageStart, getPageEnd and getFile to initiate the python scraping.
- output: This function returns a 2D array of table that was obtained from scraping.
- exception: none

render():

- output: This function renders the output of the scraping as a table.
- exception: none

17 image Module

17.1 Template Module extends React.Component

image

17.2 Uses

None.

17.3 Description

This module represents an image on the page. It details the suite of functions that are able to be executed on the image: adding the image and rendering it. It is therefore a controller module.

17.4 Syntax

17.4.1 Exported Constants

None.

17.4.2 Exported Types

ReactDOM

17.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor	HTML attribute		
getPictureLocation			
render		ReactDOM	

17.5 Semantics

17.5.1 State Variables

props : HTML attribute
numPages : Number
offset : Number

17.5.2 State Invariant

None

17.5.3 Assumptions

None.

17.5.4 Access Routine Semantics

constructor(*props*):

- transition: $state = initialBlock$
- output: $out :=$ self
- exception: none

getImageLocation():

- output: This function obtains the image location from the User
- exception: none

render():

- output: This function renders the output as an image component.
- exception: none

18 PDFScraper Module

18.1 Template Module

PDFScraper

18.2 Uses

DateTime, Tabula-py, Pandas

18.3 Description

This module handles obtaining date and task information from a PDF file.

18.4 Syntax

18.4.1 Exported Constants

None

18.4.2 Exported Types

2D List of (String, String)

18.4.3 Exported Access Programs

Routine name	In	Out	Exceptions
constructor			
importFile	String		ValueError
isDateHeading	String	Boolean	
getDataFrames	Number, Number	List of DataFrames	ValueError
getDeadlines	List of DataFrames	Number	
generateOutput	List of DataFrames, Number	2D List of (String, String)	
scrape	String, Number, Number	2D List of (String, String)	

18.5 Semantics

18.5.1 State Variables

dateStrings : List of String
dates : List of String
tasks : List of String *filePath* : String

18.5.2 State Invariant

None

18.5.3 Assumptions

It is assumed that a single instance of PDFScraper will be used.

18.5.4 Access Routine Semantics

constructor():

- transition: *dateStrings* := Appropriate date identifiers, *dates* := Empty list, *tasks* := Empty list, *filePath* := null
- output: *out* := self
- exception: None

importFile(filePath):

- transition: *filePath* := filePath
- output: None
- exception: *exc* := filePath does not end in .pdf ⇒ *ValueError*

isDateHeading(heading):

- transition: None
- output: *out* := *heading* is a valid date identifier
- exception: None

getDataFrames(startPage, endPage):

- transition: None
- output: This function returns a list of DataFrame objects corresponding to tabular data from the PDF file for only the specified page range.
- exception: $exc := (startPage < 1) \vee (startPage > endPage) \Rightarrow ValueError$

getDeadlines(dfs):

- transition: *dates* := Scraped dates, *tasks* := Scraped tasks
- output: This function returns the number of date tables found based on the list of data frames: *out* := numDateTables. It should use the isDateHeading() access routine to test potential identifiers.
- exception: None

generateOutput(dfs, numDateTables):

- transition: *dates* := Scraped dates, *tasks* := Scraped tasks
- output: This function returns a 2D list in the form [(date1, task1), (date2, task2), ...] using the updated *dates* and *tasks* state variables.
- exception: None

scrape(filePath, startPage, endPage):

- transition: None
- output: This function uses the importFile(), getDataFrames(), getDeadlines(), and generateOutput() access routines together to determine and output a 2D list in the form [(date1, task1), (date2, task2), ...].
- exception: None