SE 3XA3: Module Interface Specification Ratava

Team 9, Makiam Group
Aidan McPhelim - mcpheima
Alexie McDonald - mcdona16
Illya Pilipenko - pilipeni

December 4, 2018

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
2018-11-08	1.0	Created initial Draft
2018-11-08	1.0	Filled in a majority of module access routines
2018-11-08	1.0	Final touches
2018-12-03	1.1	Updated team name

GenerateHash Module

Module

GenerateHash

Uses

N/A

Syntax

Exported Types

None

Exported Access Programs

Routine name	In	Out	Exceptions
generate_hash	Str, Str	seq of hex	None

Semantics

State Variables

None

State Invariant

None

Assumptions

The hash type the user will pass to the retrieve hash algorithm function will be a hashtype supported by the python hashlib library

Access Routine Semantics

generate_hash($hash\_input, hash\_type$):

- output: $out := retrieve\_hash\_algorithm(hash\_type)(bytes(hash\_input))$

- exception: None

Local Functions

retrieve_hash_function: $Str \rightarrow hashfunction : seqof 0, 1 \rightarrow seqofhex$
retrieve_hash_function(type) $hashfunction : type$

bytes: $str \rightarrow seqof 0, 1$
bytes(string) $\equiv bytesofUTF-8encodingofstring$

UseHash Module

Module

UseHash

Uses

GenerateHash

Syntax

Exported Types

None

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
hash_to_colours	Str	seq of (tuple of $\mathbb{Z}$)	

Semantics

State Variables

COLOUR_AMOUNT: $\mathbb{Z}$
HEX_COLOUR_LEN: $\mathbb{Z}$
HEX_BASE: $\mathbb{Z}$
MINIMUM_COLOUR_HASH_LEN: $\mathbb{Z}$

State Invariant

None

Assumptions

None

Access Routine Semantics

hash_to_colours($hash\_value$):

Condition	$out :=$
$\lvert hash\_value \rvert \geq MINIMUM\_COLOUR\_HASH\_LEN$	$(i \lvert i \in split\_hash(hash\_value, HEX\_COLOUR\_LEN, COLOUR\_AMOUNT) \bullet hex\_to\_rgb(i))$
$\lvert hash\_value \rvert < MINIMUM\_COLOUR\_HASH\_LEN$	$(i \lvert i \in split\_hash(missing\_value(hash\_value, HEX\_COLOUR\_LEN, COLOUR\_AMOUNT)) \bullet hex\_to\_rgb(i))$

- exception: None

Local Functions

split_hash: $str \times \mathbb{Z} \times \mathbb{Z} \rightarrow seqofStr$
split_hash(hash_value, length, amount) $\equiv tokens : (\forall i \mid 0 \leq i < amount \bullet tokens(i) = hash\_value[i * length : (i+1) * length])$

hex_to_rgb: $hex \rightarrow tuple(\mathbb{Z}, \mathbb{Z}, \mathbb{Z})$
hex_to_rgb(hex_colour) $\equiv (\lvert hex\_colour \rvert \neq HEX\_COLOUR\_LEN \Rightarrow 0 \lvert \lvert hex\_colour \rvert = HEX\_COLOUR\_LEN \Rightarrow tuple(hex\_colour[1:4], hex\_colour[2:5], hex\_colour[3:6])$

missing_value: $str \times \mathbb{Z} \rightarrow str$
missing_value(hash_value) $\equiv hash\_value + hash\_value[: MINIMUM\_COLOUR\_LEN - \lvert hash\_value \rvert]$

GraphicsDraw Module

Module

GraphicsDraw

Uses

TemplateDraw, UseHash

Syntax

Exported Types

None

Exported Access Programs

Routine name	**In**	**Out**
generateColours	$\mathbb{Z}$, Str	seq of Str
fetch_template	Str	seq of (seq of (tuple of $\mathbb{Z}$
draw_image	seq (seq of (seq of (tuple of $\mathbb{Z}, \mathbb{Z}, \mathbb{Z}, \mathbb{Z}$))), seq of Str	
show_image		
save_image	Str	

Semantics

State Variables

OUTPUT_IMAGE: seq of (tuple of ($\mathbb{Z}, \mathbb{Z}, \mathbb{Z}, \mathbb{Z}, Str$))

State Invariant

None

Assumptions

None

Access Routine Semantics

generateColours($number\_of\_colours, user\_string$):

- output: $out := hash\_to\_colours(userstring)$
- exception: None

fetch_template($template\_name$):

- output: $out := generate\_template(template\_name)$
- exception: None

show_image():

- transition: OUTPUT_IMAGE will show in an image editor window
- output: None
- exception: None

draw_image($template\_data, colours$):

- transition: $OUTPUT\_IMAGE := (i|0 \leq i < |template\_data| \bullet$ $(template\_data[0][i][0], template\_data[0][i][1], template\_data[0][i][2],$ $template\_data[0][i][3], colours[0])$ $+(i|0 \leq i < |template\_data| \bullet (template\_data[1][i][0], template\_data[1][i][1],$ $template\_data[1][i][2], template\_data[1][i][3], colours[1])$
- output: None
- exception: None

save_image($location$):

- transition: OUTPUT_IMAGE will be saved to *location*
- output: None
- exception: None

Local Functions

None

TemplateDraw Module

Module

TemplateDraw

Uses

None

Syntax

Exported Types

None

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
generate_template	Str	seq of (seq of (seq of (tuple of $\mathbb{Z}, \mathbb{Z}, \mathbb{Z}, \mathbb{Z}$)))	

Semantics

State Variables

PERSON_TEMPLATE: seq of (seq of (seq of (tuple of $\mathbb{Z}, \mathbb{Z}, \mathbb{Z}, \mathbb{Z}$)))
DOG_TEMPLATE: seq of (seq of (seq of (tuple of $\mathbb{Z}, \mathbb{Z}, \mathbb{Z}, \mathbb{Z}$)))

State Invariant

None

Assumptions

None

Access Routine Semantics

generate_template(*template_name*):

- output: $out := (template_name = $ "Person" $\Rightarrow PERSON_TEMPLATE | template_name = $ "Dog" $\Rightarrow DOG\_TEMPLATE)$
- exception: None

Local Functions

None

GUI Module

Module

GUI

Uses

GraphicsDraw

Syntax

Exported Types

None

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
generate_button_clicked			
fix_body_colour			
fix_accessory_colour			
fix_template			

Semantics

State Variables

COLOURS: seq of Str
TEMPLATE: Str

State Invariant

None

Assumptions

None

Access Routine Semantics

generate_button_clicked():

- transition: draw_image(fetch_template(TEMPLATE), generateColours(COLOURS))
- output: None
- exception: None

fix_body_colour():

- transition: COLOURS[0] := text_input_1_value
- output: None
- exception: None

fix_accessory_colour():

- transition: COLOURS[1] := text_input_2_value
- output: None
- exception: None

fix_template():

- transition: (person_template = true ⇒ TEMPLATE := "Person" — dog_template = true ⇒ TEMPLATE := "Dog")
- output: None
- exception: None