

RadiPOP_API

Generated by Doxygen 1.9.2

1 Namespace Index	1
1.1 Packages	1
2 File Index	3
2.1 File List	3
3 Namespace Documentation	5
3.1 run_app Namespace Reference	5
3.1.1 Function Documentation	6
3.1.1.1 correctPartition()	6
3.1.1.2 drawOnMask()	6
3.1.1.3 extendThresholds()	7
3.1.1.4 getMask()	7
3.1.1.5 highlightOrgan()	8
3.1.1.6 initialize()	8
3.1.1.7 labelOrgan()	9
3.1.1.8 postPickleGetMask()	9
3.1.1.9 saveMasks()	9
3.1.1.10 updateMask()	10
3.1.2 Variable Documentation	10
3.1.2.1 app	10
3.1.2.2 FLASK_PORT	11
3.1.2.3 FLAST_HOST	11
3.1.2.4 host	11
3.1.2.5 patients	11
3.1.2.6 port	11
4 File Documentation	13
4.1 /Users/lorenz/Desktop/RadiPOP-standalone/web_app/run_app.py File Reference	13
Index	15

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

run_app	5
-----------------------------------	---

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/Users/lorenz/Desktop/RadiPOP-standalone/web_app/[run_app.py](#) 13

Chapter 3

Namespace Documentation

3.1 run_app Namespace Reference

Functions

- def [correctPartition](#) ()
Reveives index to slice/mask + coordinates--> returns partion corrected mask as PNG to client.
- def [drawOnMask](#) ()
Reveives index to slice/mask + x,y coordinates --> returns drawn on mask as PNG to client.
- def [extendThresholds](#) ()
Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.
- def [getMask](#) ()
Reveives index to slice/mask --> returns mask stored on flask server as PNG to client.
- def [highlightOrgan](#) ()
Reveives index of slice + x,y coordinates --> returns highlighted mask as PNG to client.
- def [initialize](#) ()
Receive Paths to ordered slices, caches slices.
- def [labelOrgan](#) ()
Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.
- def [postPickleGetMask](#) ()
Receives path to pickle file --> returns mask as PNG to client.
- def [saveMasks](#) ()
Reveives path, saves all stored masks as pickle files to path --> returns output path.
- def [updateMask](#) ()
Receives index of slice + slider values --> returns updated mask as PNG to client.

Variables

- [app](#) = Flask(__name__)
- int [FLASK_PORT](#) = 4041
- string [FLASK_HOST](#) = '0.0.0.0'
- [host](#)
- dictionary [patients](#) = {}
Dictionary which will hold for each patientID a RadiPopGUI object.
- [port](#)

3.1.1 Function Documentation

3.1.1.1 correctPartition()

```
def run_app.correctPartition ( )
```

Reveives index to slice/mask + coordinates--> returns partion corrected mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>coordinates</i>	array of coordinates of the form [x0,y0,x1,y1,...,xn,yn]

Returns

mask as transparent PNG as byte stream

Note: The coordinates array will be used to generate a line that cuts/divides the segmented organs.

Example handling of return image stream in js:

```
bytestring = data['status']
img = bytestring.split('\')[1]
target.src = "data:image/png;base64," + img;
```

3.1.1.2 drawOnMask()

```
def run_app.drawOnMask ( )
```

Reveives index to slice/mask + x,y coordinates --> returns drawn on mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>coordinates</i>	array of coordinates of the form [x0,y0,x1,y1,...,xn,yn]

Returns

mask as transparent PNG as byte stream

Note: The coordinates array will be used to draw a line on the mask.

Example handling of return image stream in js:

```
bytestring = data['status']  
img = bytestring.split('\n')[1]  
target.src = "data:image/png;base64," + img;
```

3.1.1.3 extendThresholds()

```
def run_app.extendThresholds ( )
```

Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>left</i>	Extend labeling up to index-label
<i>right</i>	Extend labeling up to index+label

Returns

json data containing left_most_idx and right_most_idx

Note: The left_most_idx and right_most_idx correspond to the indices of the slices up to which the labeling has been extended. After the the function has finished use the function API's function /getMask to update the masks in your GUI. Example in js:

```
for (let index=parseInt(data["left_most_idx"]); index<parseInt(data["right_most_idx"])+1; index++) {  
    $.post(FLASK_SERVER+"/getMask", {  
        javascript_data: JSON.stringify({patienID: id, index: idx})  
    })  
}
```

3.1.1.4 getMask()

```
def run_app.getMask ( )
```

Reveives index to slice/mask --> returns mask stored on flask server as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated

Returns

mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
bytestring = data['status']
img = bytestring.split('\n')[1]
target.src = "data:image/png;base64," + img;
```

3.1.1.5 highlightOrgan()

```
def run_app.highlightOrgan ( )
```

Reveives index of slice + x,y coordinates --> returns highlighted mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>x</i>	relative x coordinates (0<=x<=1)
<i>y</i>	relative y coordinates (0<=y<=1)

Returns

mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
bytestring = data['status']
img = bytestring.split('\n')[1]
target.src = "data:image/png;base64," + img;
```

3.1.1.6 initialize()

```
def run_app.initialize ( )
```

Receive Paths to ordered slices, caches slices.

Parameters

<i>patientID</i>	The ID of the patient
<i>paths</i>	An array with the paths to the slices

Returns

200,OK

Note: Paths to slices !!!MUST BE ORDERED!!! 0,1,...,n

3.1.1.7 labelOrgan()

```
def run_app.labelOrgan ( )
```

Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>label</i>	Label of organ (1 for liver, 2 for spleen, 0 nothing, >2 other organ)

Returns

mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
bytestring = data['status']  
img = bytestring.split('\')[1]  
target.src = "data:image/png;base64," + img;
```

3.1.1.8 postPickleGetMask()

```
def run_app.postPickleGetMask ( )
```

Receives path to pickle file --> returns mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice the mask refers to
<i>path</i>	The path to the mask file

Returns

mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
bytestring = data['status']  
img = bytestring.split('\')[1]  
target.src = "data:image/png;base64," + img;
```

3.1.1.9 saveMasks()

```
def run_app.saveMasks ( )
```

Reveives path, saves all stored masks as pickle files to path --> returns output path.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated

Returns

path/directory to which the pickle files were written

3.1.1.10 updateMask()

```
def run_app.updateMask ( )
```

Receives index of slice + slider values --> returns updated mask as PNG to client.

Parameters

<i>patientID</i>	The ID of the patient
<i>index</i>	The index of the slice for which to mask should be updated
<i>liver-intensity-slider</i>	Slider value for liver intensity
<i>bone-intensity-slider</i>	Slider value for bone intensity
<i>blood-vessel-intensity-slider</i>	Slider value for blood-vessel intensity

Returns

mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
bytestring = data['status']
img = bytestring.split('\n')[1]
target.src = "data:image/png;base64," + img;
```

3.1.2 Variable Documentation**3.1.2.1 app**

```
app = Flask(__name__)
```

3.1.2.2 FLASK_PORT

```
int FLASK_PORT = 4041
```

3.1.2.3 FLAST_HOST

```
FLAST_HOST = '0.0.0.0'
```

3.1.2.4 host

```
host
```

3.1.2.5 patients

```
dictionary patients = {}
```

Dictionary which will hold for each patientID a RadiPopGUI object.

Patients are added by the API's /initialize function

3.1.2.6 port

```
port
```


Chapter 4

File Documentation

4.1 /Users/lorenz/Desktop/RadiPOP-standalone/web_app/run_app.py File Reference

Namespaces

- namespace `run_app`

Functions

- def `correctPartition ()`
Reveives index to slice/mask + coordinates--> returns partion corrected mask as PNG to client.
- def `drawOnMask ()`
Reveives index to slice/mask + x,y coordinates --> returns drawn on mask as PNG to client.
- def `extendThresholds ()`
Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.
- def `getMask ()`
Reveives index to slice/mask --> returns mask stored on flask server as PNG to client.
- def `highlightOrgan ()`
Reveives index of slice + x,y coordinates --> returns highlighted mask as PNG to client.
- def `initialize ()`
Receive Paths to ordered slices, caches slices.
- def `labelOrgan ()`
Reveives index to slice mask + label id --> returns highlighted mask as PNG to client.
- def `postPickleGetMask ()`
Receives path to pickle file --> returns mask as PNG to client.
- def `saveMasks ()`
Reveives path, saves all stored masks as pickle files to path --> returns output path.
- def `updateMask ()`
Receives index of slice + slider values --> returns updated mask as PNG to client.

Variables

- `app = Flask(__name__)`
- int `FLASK_PORT = 4041`
- string `FLASK_HOST = '0.0.0.0'`
- `host`
- dictionary `patients = {}`
Dictionary which will hold for each patientID a RadiPopGUI object.
- `port`

Index

/Users/lorenz/Desktop/RadiPOP-standalone/web_app/run_app.py, 13

app
run_app, 10

correctPartition
run_app, 6

drawOnMask
run_app, 6

extendThresholds
run_app, 7

FLASK_PORT
run_app, 10

FLAST_HOST
run_app, 11

getMask
run_app, 7

highlightOrgan
run_app, 8

host
run_app, 11

initialize
run_app, 8

labelOrgan
run_app, 8

patients
run_app, 11

port
run_app, 11

postPickleGetMask
run_app, 9

run_app, 5
app, 10
correctPartition, 6
drawOnMask, 6
extendThresholds, 7
FLASK_PORT, 10
FLAST_HOST, 11
getMask, 7
highlightOrgan, 8
host, 11
initialize, 8
labelOrgan, 8
patients, 11
port, 11
postPickleGetMask, 9
saveMasks, 9
updateMask, 10

saveMasks
run_app, 9

updateMask
run_app, 10