

RadiPOP\_API

Generated by Doxygen 1.9.2



<b>1 Namespace Index</b>	<b>1</b>
1.1 Packages	1
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>7</b>
4.1 File List	7
<b>5 Namespace Documentation</b>	<b>9</b>
5.1 config Namespace Reference	9
5.2 radipop_gui Namespace Reference	9
5.3 segmentation_utils Namespace Reference	9
5.3.1 Function Documentation	10
5.3.1.1 add_sobel_edges()	10
5.3.1.2 draw_region_outlines()	10
5.3.1.3 guess_bounds()	11
5.3.1.4 partition_at_threshold()	11
5.3.1.5 save_partition()	12
5.4 segmenter_flask_API Namespace Reference	12
5.4.1 Function Documentation	13
5.4.1.1 correctPartition()	13
5.4.1.2 dcm2png()	13
5.4.1.3 dcm2pngPreview()	14
5.4.1.4 drawOnMask()	14
5.4.1.5 extendLabels()	15
5.4.1.6 getMask()	15
5.4.1.7 highlightOrgan()	16
5.4.1.8 initialize()	16
5.4.1.9 labelOrgan()	17
5.4.1.10 postPickleGetMask()	17
5.4.1.11 saveMasks()	18
5.4.1.12 updateMask()	18
5.4.2 Variable Documentation	19
5.4.2.1 app	19
5.4.2.2 FLASK_PORT	19
5.4.2.3 FLAST_HOST	19
5.4.2.4 host	19
5.4.2.5 patients	19
5.4.2.6 port	19

<b>6 Class Documentation</b>	<b>21</b>
6.1 BaseConfig Class Reference	21
6.1.1 Member Data Documentation	21
6.1.1.1 SECRET_KEY	21
6.2 DevelopmentConfig Class Reference	21
6.2.1 Member Data Documentation	22
6.2.1.1 DEBUG	22
6.3 ProductionConfig Class Reference	22
6.4 RadiPopGUI Class Reference	22
6.4.1 Detailed Description	24
6.4.2 Constructor & Destructor Documentation	24
6.4.2.1 __init__()	24
6.4.3 Member Function Documentation	25
6.4.3.1 clip_dcm()	25
6.4.3.2 color_mask()	25
6.4.3.3 correct_partition()	26
6.4.3.4 create_image_stream()	26
6.4.3.5 draw_on_image()	26
6.4.3.6 extend_labels()	27
6.4.3.7 extract_metadata_from_dcm()	27
6.4.3.8 find_organs()	27
6.4.3.9 highlight_np_label_array_to_png()	29
6.4.3.10 highlightOrgan()	30
6.4.3.11 labelMask()	30
6.4.3.12 np_label_array_to_png()	30
6.4.3.13 read_pickle_mask_to_np_label_array()	31
6.4.3.14 readPNG()	31
6.4.3.15 save_masks()	32
6.4.3.16 slice_dim()	32
6.4.3.17 update_mask_upon_slider_change()	32
6.4.3.18 writePillow2PNG()	33
6.4.4 Member Data Documentation	33
6.4.4.1 LABELS	33
6.4.4.2 last_clicked_x	33
6.4.4.3 last_clicked_y	33
6.4.4.4 LIVER_LABEL	34
6.4.4.5 masks	34
6.4.4.6 pathToSlices	34
6.4.4.7 patient_id	34
6.4.4.8 selected_pixel_value_of_label_mask	34
6.4.4.9 sliceCache	34
6.4.4.10 SPLEEN_LABEL	35

---

6.5 TestingConfig Class Reference . . . . .	35
6.5.1 Member Data Documentation . . . . .	35
6.5.1.1 DEBUG . . . . .	35
<b>7 File Documentation</b>	<b>37</b>
7.1 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/segmenter_flask_API.py File Reference . . . . .	37
7.2 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/config.py File Reference	38
7.3 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/radipop_gui.py File Reference . . . . .	38
7.4 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/segmentation_utils.py File Reference . . . . .	38
<b>Index</b>	<b>41</b>



# Chapter 1

## Namespace Index

### 1.1 Packages

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

<a href="#">config</a>	9
<a href="#">radipop_gui</a>	9
<a href="#">segmentation_utils</a>	9
<a href="#">segmenter_flask_API</a>	12





## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

BaseConfig . . . . .	<a href="#">21</a>
DevelopmentConfig . . . . .	<a href="#">21</a>
ProductionConfig . . . . .	<a href="#">22</a>
TestingConfig . . . . .	<a href="#">35</a>
RadiPopGUI . . . . .	<a href="#">22</a>



## Chapter 3

# Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">BaseConfig</a> . . . . .	21
<a href="#">DevelopmentConfig</a> . . . . .	21
<a href="#">ProductionConfig</a> . . . . .	22
<a href="#">RadiPopGUI</a>	
Bridge between the flask server/API and the RadiPOP segmenter: . . . . .	22
<a href="#">TestingConfig</a> . . . . .	35



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/ <a href="#">segmenter_flask_API.py</a> . . .	37
/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/ <a href="#">config.py</a> . . . . .	38
/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/ <a href="#">radipop_gui.py</a> . . . . .	38
/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/ <a href="#">segmentation_utils.py</a> . .	38



## Chapter 5

# Namespace Documentation

### 5.1 config Namespace Reference

#### Classes

- class [BaseConfig](#)
- class [DevelopmentConfig](#)
- class [ProductionConfig](#)
- class [TestingConfig](#)

### 5.2 radipop\_gui Namespace Reference

#### Classes

- class [RadiPopGUI](#)

*Bridge between the flask server/API and the RadiPOP segmenter:*

### 5.3 segmentation\_utils Namespace Reference

#### Functions

- def [add\\_sobel\\_edges](#) (mask, img)  
*Smooth edges Steps:*
- def [draw\\_region\\_outlines](#) (mask)  
*Color the mask light green.*
- def [guess\\_bounds](#) (regions\_map, reference\_map)  
*Guess the bounds/labels of the region based on reference region Guess the bounds/labels of the region based on reference region (generally neighboring slice).*
- def [partition\\_at\\_threshold](#) (img, thresh, square\_size, min\_size, title=None, show\_plot=True)  
*After some smoothing, calculate new mask for img Steps:*
- def [save\\_partition](#) (mask, path)

### 5.3.1 Function Documentation

#### 5.3.1.1 `add_sobel_edges()`

```
def segmentation_utils.add_sobel_edges (
    mask,
    img )
```

Smooth edges Steps:

- Edge filter image using the Canny algorithm.
- euclidean distance transform

##### Parameters

<i>mask</i>	mask corresponding to image
<i>img</i>	image corresponding to mask

##### Returns

mask with smoothed edges

#### 5.3.1.2 `draw_region_outlines()`

```
def segmentation_utils.draw_region_outlines (
    mask )
```

Color the mask light green.

Color the edges of the mask darker green.

##### Parameters

<i>mask</i>	mask for which to color the outlines
-------------	--------------------------------------

##### Returns

mask with colored outlines



### 5.3.1.3 guess\_bounds()

```
def segmentation_utils.guess_bounds (
    regions_map,
    reference_map )
```

Guess the bounds/labels of the region based on reference region Guess the bounds/labels of the region based on reference region (generally neighboring slice).

#### Parameters

<i>regions_map</i>	mask to guess labels for
<i>reference_map</i>	reference mask (already labelled)

#### Returns

mask (labelled)

### 5.3.1.4 partition\_at\_threshold()

```
def segmentation_utils.partition_at_threshold (
    img,
    thresh,
    square_size,
    min_size,
    title = None,
    show_plot = True )
```

After some smoothing, calculate new mask for img Steps:

- gaussian filter,
- remove small objects,
- greyscale morphological closing,
- euclidean distance transform

#### Parameters

<i>img</i>	type numpy.ndarray
<i>thres</i>	Threshold value
<i>min_size</i>	Minimum size of an organ in the mask
<i>squaresize</i>	For greyscale morphological closing
<i>title</i>	Title of plot
<i>show_plot</i>	Show plot True/False

**Returns**

New binary mask (same size as img)

**5.3.1.5 save\_partition()**

```
def segmentation_utils.save_partition (
    mask,
    path )
```

**5.4 segmenter\_flask\_API Namespace Reference****Functions**

- def [correctPartition](#) ()  
*Reveives index to slice/mask + coordinates--> returns partion corrected mask as PNG to client.*
- def [dcm2png](#) ()  
*Receive Paths to dcm files, converts them to PNG.*
- def [dcm2pngPreview](#) ()  
*Receive Paths to dcm file, converts it to PNG.*
- def [drawOnMask](#) ()  
*Reveives index to slice/mask + x,y coordinates --> returns drawn on mask as PNG to client.*
- def [extendLabels](#) ()  
*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](#)

## 5.4.1 Function Documentation

### 5.4.1.1 correctPartition()

```
def segmenter_flask_API.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

JSON: {mask: byte stream} 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:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\')[1]
```

### 5.4.1.2 dcm2png()

```
def segmenter_flask_API.dcm2png ( )
```

Receive Paths to dcm files, converts them to PNG.

Included metadata IDs: "PatientID","PatientBirthDate","PatientName", "PatientAge","PatientSex","PatientName","↵  
SliceThickness","StudyID","ContentDate"

#### Parameters

<i>paths</i>	An array with the paths to the slices
<i>low_clip</i>	lowest pixel value (Recommended:850)
<i>high_clip</i>	highest pixel value (Recommended: 1250)

**Returns**

JSON: {message: message, metadata: dictionary}

**5.4.1.3 dcm2pngPreview()**

```
def segmenter_flask_API.dcm2pngPreview ( )
```

Receive Paths to dcm file, converts it to PNG.

Included metadata IDs: "PatientID", "PatientBirthDate", "PatientName", "PatientAge", "PatientSex", "PatientName", "← SliceThickness", "StudyID", "ContentDate"

**Parameters**

<i>path</i>	The paths to the dcm file
<i>low_clip</i>	lowest pixel value (Recommended:850)
<i>high_clip</i>	highest pixel value (Recommended: 1250)

**Returns**

JSON: {slice: slice stream} slice as PNG as byte stream

**5.4.1.4 drawOnMask()**

```
def segmenter_flask_API.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**

JSON: {mask: byte stream} 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:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
```

```

    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(data)
  })
  .then(function(response) { return response.json(); })
  .then(function(data) {
    bytestring = data["mask"];
    img = bytestring.split('\')[1]
  })

```

#### 5.4.1.5 extendLabels()

```
def segmenter_flask_API.extendLabels ( )
```

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-left
<i>right</i>	Extend labeling up to index+right

##### Returns

JSON: {left\_most\_idx: idx, right\_most\_idx: 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})
  })
}

```

#### 5.4.1.6 getMask()

```
def segmenter_flask_API.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**

JSON: {mask: byte stream} mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\')[1]
```

**5.4.1.7 highlightOrgan()**

```
def segmenter_flask_API.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**

JSON: {mask: byte stream} mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\')[1]
```

**5.4.1.8 initialize()**

```
def segmenter_flask_API.initialize ( )
```

Receive Paths to ordered slices, caches slices.

## Parameters

<i>patientID</i>	The ID of the patient. Must be unique!
<i>paths</i>	An array with the paths to the slices

## Returns

JSON: {message: message}

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

## 5.4.1.9 labelOrgan()

```
def segmenter_flask_API.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

JSON: {mask: byte stream} mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\')[1]
```

## 5.4.1.10 postPickleGetMask()

```
def segmenter_flask_API.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**

JSON: {mask: byte stream} mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\')[1]
```

**5.4.1.11 saveMasks()**

```
def segmenter_flask_API.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**

JSON: {outdir: path} path/directory to which the pickle files were written

**5.4.1.12 updateMask()**

```
def segmenter_flask_API.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 intesity
<i>bone-intensity-slider</i>	Slider value for bone intesity
<i>blood-vessel-intensity-slider</i>	Slider value for blood-vessel intesity



### Returns

JSON: {mask: byte stream} mask as transparent PNG as byte stream

Example handling of return image stream in js:

```
fetch(RadiPOP_states.FLASK_SERVER+"/labelOrgan", {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
})
.then(function(response){ return response.json();})
.then(function(data){
  bytestring = data["mask"];
  img = bytestring.split('\n')[1]
```

## 5.4.2 Variable Documentation

### 5.4.2.1 app

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

### 5.4.2.2 FLASK\_PORT

```
int FLASK_PORT = 4041
```

### 5.4.2.3 FLAST\_HOST

```
FLAST_HOST = '0.0.0.0'
```

### 5.4.2.4 host

```
host
```

### 5.4.2.5 patients

```
dictionary patients = {}
```

Dictionary which will hold for each patientID a RadiPopGUI object.

Patients are added by the API's /initialize function

### 5.4.2.6 port

```
port
```

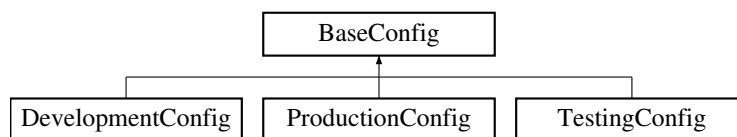


## Chapter 6

# Class Documentation

### 6.1 BaseConfig Class Reference

Inheritance diagram for BaseConfig:



#### Static Public Attributes

- `SECRET_KEY` = `os.getenv('SECRET_KEY', 'REPLACE ME')`

#### 6.1.1 Member Data Documentation

##### 6.1.1.1 SECRET\_KEY

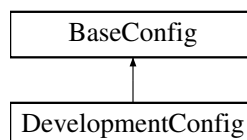
```
SECRET_KEY = os.getenv('SECRET_KEY', 'REPLACE ME') [static]
```

The documentation for this class was generated from the following file:

- `/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter_flask_API/utility/config.py`

### 6.2 DevelopmentConfig Class Reference

Inheritance diagram for DevelopmentConfig:



## Static Public Attributes

- bool `DEBUG` = True

### 6.2.1 Member Data Documentation

#### 6.2.1.1 DEBUG

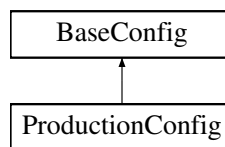
```
bool DEBUG = True [static]
```

The documentation for this class was generated from the following file:

- [/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\\_flask\\_API/utility/config.py](#)

## 6.3 ProductionConfig Class Reference

Inheritance diagram for ProductionConfig:



### Additional Inherited Members

The documentation for this class was generated from the following file:

- [/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\\_flask\\_API/utility/config.py](#)

## 6.4 RadiPopGUI Class Reference

Bridge between the flask server/API and the RadiPOP segmenter:

## Public Member Functions

- def `__init__` (self, `patient_id`)  
*Class constructor:*
- def `extend_labels` (self, `cur_idx`, `left_extend`, `right_extend`)  
*Extend labels from current slice to neighbouring slices.*
- def `highlight_np_label_array_to_png` (self, `mask_np_array`, `highlight`)  
*Takes an numpy label array  $\text{dim}(n,m,1)$  and returns a RGBA pillow image  $\text{dim}(n,m,4)$*
- def `highlightOrgan` (self, `slice_idx`, `x`, `y`)  
*Highlights regions of the mask (organs) that were clicked on by user.*
- def `labelMask` (self, `slice_idx`, `newlabel`)  
*Labels mask at given index at previously selected region.*
- def `save_masks` (self, `path`)  
*Saves masks as pickle file to given path.*
- def `slice_dim` (self, `index`)  
*Returns dimensions of slice images (x,y)*

## Static Public Member Functions

- def `clip_dcm` (`dcm_file`, `clip_low`=850, `clip_high`=1250)  
*Read dicom image (.dcm), clips it and returns it as a grey scale PNG.*
- def `color_mask` (`mask_np_array`)  
*Takes an numpy label array  $\text{dim}(n,m,1)$  and returns a color mask  $\text{dim}(n,m,4)$*
- def `correct_partition` (`image`)  
*Convert PNG mask to 1 channelled label mask.*
- def `create_image_stream` (`img`)  
*Returns base64 bytestream for given input image.*
- def `draw_on_image` (`coordinates`, `img`, `correctionMode`=False)  
*Draws a point or lines on given PNG image.*
- def `extract_metadata_from_dcm` (`dcm_file`)  
*Read dicom image (.dcm) and extract metadata Extracted metadata IDs: "PatientID", "PatientBirthDate", "PatientName", "PatientAge", "PatientSex", "PatientName", "SliceThickness", "StudyID", "ContentDate".*
- def `find_organs` (`img`, `bones_thresh`, `blood_vessels_thresh`, `liver_thresh`)  
*Uses three threshold values to find organs.*
- def `np_label_array_to_png` (`mask_np_array`)  
*Takes an numpy label array  $\text{dim}(n,m,1)$  and returns a RGBA pillow image  $\text{dim}(n,m,4)$*
- def `read_pickle_mask_to_np_label_array` (`path`)  
*Opens mask pickle file and returns it as a np.array.*
- def `readPNG` (`path`)  
*Reads an image (e.g.*
- def `update_mask_upon_slider_change` (`image`, `bone_intensity`, `blood_vessel_intensity`, `liver_intensity`)  
*Sets threshold for liver intensity.*
- def `writePillow2PNG` (`img`, `outfile`)

## Public Attributes

- [last\\_clicked\\_x](#)  
*The x-coordinate of the region that was last selected/clicked on.*
- [last\\_clicked\\_y](#)  
*The y-coordinate of the region that was last selected/clicked on.*
- [masks](#)  
*Dictionary: key: mask index, value: mask numpy array dim(n,m,1)*
- [pathToSlices](#)  
*List containing the path to the png slice files*
- [patient\\_id](#)  
*ID of the patient.*
- [selected\\_pixel\\_value\\_of\\_label\\_mask](#)  
*The label of the region that was last selected/clicked on.*
- [sliceCache](#)  
*Dictionary: key: slice index, value: slice PNG.*

## Static Public Attributes

- list [LABELS](#) = [[SPLEEN\\_LABEL](#),[SPLEEN\\_LABEL](#)]  
*list with labels*
- int [LIVER\\_LABEL](#) = 1  
*Regions in mask with this label value are considered liver.*
- int [SPLEEN\\_LABEL](#) = 2  
*Regions in mask with this label value are considered spleen.*

### 6.4.1 Detailed Description

Bridge between the flask server/API and the RadiPOP segmenter:

Note:

- For each patient one object of this class should be instantiated.
- An object of this class contains all the slices and masks associated with the patient
- This class also contains static utility functions
- This class is the bridge between the flask server/API and the RadiPOP segmenter (segmentations\_utils)

### 6.4.2 Constructor & Destructor Documentation

#### 6.4.2.1 `__init__()`

```
def __init__ (
    self,
    patient_id )
```

Class constructor:

## Parameters

<i>patient</i> ↔ _id	The ID of the patient. Must be unique!
-------------------------	--

Note:

- For each patient one object of this class should be instantiated.
- An object of this class contains all the slices and masks associated with the patient

### 6.4.3 Member Function Documentation

#### 6.4.3.1 clip\_dcm()

```
def clip_dcm (
    dcm_file,
    clip_low = 850,
    clip_high = 1250 ) [static]
```

Read dicom image (.dcm), clips it and returns it as a grey scale PNG.

```
@param dcm_file Path to .dcm file
@param clip_low lowest pixel value
@param clip_high highest pixel value

@return tuple(L (grey scale) Pillow Image, slice index)
```

#### 6.4.3.2 color\_mask()

```
def color_mask (
    mask_np_array ) [static]
```

Takes an numpy label array dim(n,m,1) and returns a color mask dim(n,m,4)

## Parameters

<i>mask_np_array</i>	numpy label array dim(n,m,1)
----------------------	------------------------------

## Returns

```
color mask image dim(n,m,4) --> RGBA

- Default liver color is red (LIVER_LABEL)
- Default spleen color is blue (SPLEEN_LABEL)
- Default for other regions is green
```

### 6.4.3.3 correct\_partition()

```
def correct_partition (
    image ) [static]
```

Convert PNG mask to 1 channelled label mask.

#### Parameters

<i>image</i>	Image (e.g.: RGBA PNG pillow)
--------------	-------------------------------

#### Returns

label mask

### 6.4.3.4 create\_image\_stream()

```
def create_image_stream (
    img ) [static]
```

Returns base64 bytestream for given input image.

#### Parameters

<i>img</i>	Image (e.g. Pillow PNG)
------------	-------------------------

#### Returns

img\_base64 stream

### 6.4.3.5 draw\_on\_image()

```
def draw_on_image (
    coordinates,
    img,
    correctionMode = False ) [static]
```

Draws a point or lines on given PNG image.

#### Parameters

<i>coordinates</i>	List of the form [x0,y0,x1,y1,...,xn,yn]
<i>img</i>	Image (e.g.: RGBA PNG pillow)
<i>correctionMode</i>	True/False If true the drawn line acts as an eraser (dividing organs). If false a colored line is drawn on the image. DEFAULT: False



The modifications are made directly on the provided image. No return value

#### 6.4.3.6 `extend_labels()`

```
def extend_labels (
    self,
    cur_idx,
    left_extend,
    right_extend )
```

Extend labels from current slice to neighbouring slices.

##### Parameters

<i>cur_idx</i>	Index of reference slice
<i>left_extend</i>	Number of slices to extend the labeling to below reference slice
<i>right_extend</i>	Number of slices to extend the labeling to above reference slice

##### Returns

(left\_most\_idx, right\_most\_idx) The index of the outermost slices to which the labeling was extended

Extends labels left and right from current slice How far the labels are extended is taken from left and right expansion bounds

#### 6.4.3.7 `extract_metadata_from_dcm()`

```
def extract_metadata_from_dcm (
    dcm_file ) [static]
```

Read dicom image (.dcm) and extract metadata Extracted metadata IDs: "PatientID", "PatientBirthDate", "PatientName", "PatientAge", "PatientSex", "PatientName", "SliceThickness", "StudyID", "ContentDate".

##### Parameters

<i>dcm_file</i>	Path to .dcm file
-----------------	-------------------

##### Returns

dictionary with metadata information

#### 6.4.3.8 `find_organs()`

```
def find_organs (
    img,
    bones_thresh,
```

```
    blood_vessels_thresh,  
    liver_thresh ) [static]
```

Uses three threshold values to find organs.

**Parameters**

<i>image</i>	Image (e.g.: RGBA PNG pillow)
<i>bones_thresh</i>	bones threshold: [threshold, square_size , min_size]
<i>blood_vessels_thresh</i>	blood vessels threshold: [threshold, square_size , min_size]
<i>liver_thresh</i>	liver threshold: [threshold, square_size , min_size]

**Returns**

New labelled mask (same size as slice)

The algorithm is:

- After some smoothing, remove every pixel above bones threshold from the image.
- After some smoothing, remove every pixel above blood vessel threshold.
- Everything that then remains above liver threshold is called an organ.
- Use contiguous area divisions to roughly split into organs.

**6.4.3.9 highlight\_np\_label\_array\_to\_png()**

```
def highlight_np_label_array_to_png (
    self,
    mask_np_array,
    highlight )
```

Takes an numpy label array dim(n,m,1) and returns a RGBA pillow image dim(n,m,4)

**Parameters**

<i>mask_np_array</i>	numpy label array dim(n,m,1) @highlight Highlight regions where highlight==label in brighter color
----------------------	--

**Returns**

RGBA pillow image dim(n,m,4)

Turns a labelled mask into a transparent (RGBA) PNG.

- Default liver color is red (LIVER\_LABEL)
- Default spleen color is blue (SPLEEN\_LABEL)
- Default for other regions is green

#### 6.4.3.10 highlightOrgan()

```
def highlightOrgan (
    self,
    slice_idx,
    x,
    y )
```

Highlights regions of the mask (organs) that were clicked on by user.

##### Parameters

<i>slice_idx</i>	Index of mask/slice to be highlighted
<i>x</i>	x-coordinate of slice (in pixels)
<i>y</i>	y-coordinate of slice (in pixels)

##### Returns

tuple(Mask where the region specified by x and y is highlighted in brigther colors, pixel\_value)

#### 6.4.3.11 labelMask()

```
def labelMask (
    self,
    slice_idx,
    newlabel )
```

Labels mask at given index at previously selected region.

##### Parameters

<i>slice_idx</i>	Index of mask/slice to be labelled
<i>label</i>	Label to be assigned to previously selected region (either LIVER_LABEL or SPLEEN_LABEL or -1 (for removing label))

##### Returns

Mask with new label

Note: It is expected that the client has before highlighted an organ with the function [self.highlightOrgan\(\)](#). This determines the region/organ that will be labelled with label.

#### 6.4.3.12 np\_label\_array\_to\_png()

```
def np_label_array_to_png (
    mask_np_array ) [static]
```

Takes an numpy label array dim(n,m,1) and returns a RGBA pillow image dim(n,m,4)

## Parameters

<i>mask_np_array</i>	numpy label array dim(n,m,1) @highlight Highlight regions where highlight==label in brighter color
----------------------	--

## Returns

RGBA pillow image dim(n,m,4)

Turns a labelled mask into a transparent (RGBA) PNG.

- Default liver color is red (LIVER\_LABEL)
- Default spleen color is blue (SPLEEN\_LABEL)
- Default for other regions is green

**6.4.3.13 read\_pickle\_mask\_to\_np\_label\_array()**

```
def read_pickle_mask_to_np_label_array (
    path ) [static]
```

Opens mask pickle file and returns it as a np.array.

## Parameters

<i>path</i>	Path to pickle file
-------------	---------------------

## Returns

numpy array of mask

**6.4.3.14 readPNG()**

```
def readPNG (
    path ) [static]
```

Reads an image (e.g.

: PNG file) to numpy array

## Parameters

<i>path</i>	Path to image
-------------	---------------

**Returns**

numpy array of image

**6.4.3.15 save\_masks()**

```
def save_masks (
    self,
    path )
```

Saves masks as pickle file to given path.

**Parameters**

<i>path</i>	Path to which mask files should be written
-------------	--

Masks are written as .p (pickle) files

**6.4.3.16 slice\_dim()**

```
def slice_dim (
    self,
    index )
```

Returns dimensions of slice images (x,y)

**Returns**

(x,y) Dimensions of slices

**6.4.3.17 update\_mask\_upon\_slider\_change()**

```
def update_mask_upon_slider_change (
    image,
    bone_intensity,
    blood_vessel_intensity,
    liver_intensity ) [static]
```

Sets threshold for liver intensity.

**Parameters**

<i>image</i>	Image (e.g.: RGBA PNG pillow)
<i>bones_thresh</i>	bones threshold
<i>blood_vessels_thresh</i>	blood vessels threshold
<i>liver_thresh</i>	liver threshold

#### Returns

New labelled mask

#### Steps:

- Sets thresholds for current slice
- Runs self.find\_organ on current slice with new thresholds

#### 6.4.3.18 writePillow2PNG()

```
def writePillow2PNG (
    img,
    outfile ) [static]
```

### 6.4.4 Member Data Documentation

#### 6.4.4.1 LABELS

```
list LABELS = [SPLEEN_LABEL, SPLEEN_LABEL] [static]
```

list with labels

#### 6.4.4.2 last\_clicked\_x

```
last_clicked_x
```

The x-coordinate of the region that was last selected/clicked on.

#### 6.4.4.3 last\_clicked\_y

```
last_clicked_y
```

The y-coordinate of the region that was last selected/clicked on.

#### 6.4.4.4 LIVER\_LABEL

```
int LIVER_LABEL = 1 [static]
```

Regions in mask with this label value are considered liver.

#### 6.4.4.5 masks

```
masks
```

Dictionary: key: mask index, value: mask numpy array dim(n,m,1)

#### 6.4.4.6 pathToSlices

```
pathToSlices
```

List containing the path to the png slice files

#### 6.4.4.7 patient\_id

```
patient_id
```

ID of the patient.

#### 6.4.4.8 selected\_pixel\_value\_of\_label\_mask

```
selected_pixel_value_of_label_mask
```

The label of the region that was last selected/clicked on.

#### 6.4.4.9 sliceCache

```
sliceCache
```

Dictionary: key: slice index, value: slice PNG.



#### 6.4.4.10 SPLEEN\_LABEL

```
int SPLEEN_LABEL = 2 [static]
```

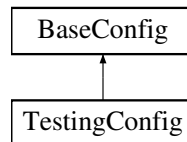
Regions in mask with this label value are considered spleen.

The documentation for this class was generated from the following file:

- [/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\\_flask\\_API/utility/radipop\\_gui.py](#)

## 6.5 TestingConfig Class Reference

Inheritance diagram for TestingConfig:



### Static Public Attributes

- bool `DEBUG` = True

### 6.5.1 Member Data Documentation

#### 6.5.1.1 DEBUG

```
bool DEBUG = True [static]
```

The documentation for this class was generated from the following file:

- [/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\\_flask\\_API/utility/config.py](#)



## Chapter 7

# File Documentation

### 7.1 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_flask\_API/segmenter\_flask\_API.py File Reference

#### Namespaces

- namespace [segmenter\\_flask\\_API](#)

#### Functions

- def [correctPartition](#) ()  
*Reveives index to slice/mask + coordinates--> returns partion corrected mask as PNG to client.*
- def [dcm2png](#) ()  
*Receive Paths to dcm files, converts them to PNG.*
- def [dcm2pngPreview](#) ()  
*Receive Paths to dcm file, converts it to PNG.*
- def [drawOnMask](#) ()  
*Reveives index to slice/mask + x,y coordinates --> returns drawn on mask as PNG to client.*
- def [extendLabels](#) ()  
*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`

## 7.2 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_↔ flask\_API/utility/config.py File Reference

### Classes

- class `BaseConfig`
- class `DevelopmentConfig`
- class `ProductionConfig`
- class `TestingConfig`

### Namespaces

- namespace `config`

## 7.3 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_↔ flask\_API/utility/radipop\_gui.py File Reference

### Classes

- class `RadiPopGUI`  
*Bridge between the flask server/API and the RadiPOP segmenter:*

### Namespaces

- namespace `radipop_gui`

## 7.4 /Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_↔ flask\_API/utility/segmentation\_utils.py File Reference

### Namespaces

- namespace `segmentation_utils`

## Functions

- def [add\\_sobel\\_edges](#) (mask, img)  
*Smooth edges Steps:*
- def [draw\\_region\\_outlines](#) (mask)  
*Color the mask light green.*
- def [guess\\_bounds](#) (regions\_map, reference\_map)  
*Guess the bounds/labels of the region based on reference region Guess the bounds/labels of the region based on reference region (generally neighboring slice).*
- def [partition\\_at\\_threshold](#) (img, thresh, square\_size, min\_size, title=None, show\_plot=True)  
*After some smoothing, calculate new mask for img Steps:*
- def [save\\_partition](#) (mask, path)



# Index

/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_flask\_API/segmenter\_flask\_API.py, 37

/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_flask\_API/utility/config.py, 38

/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_flask\_API/utility/radipop\_gui.py, 38

/Users/lorenz/Desktop/temp/radipop/electron-react/segmenter\_flask\_API/utility/segmentation\_utils.py, 38

\_\_init\_\_  
RadiPopGUI, 24

add\_sobel\_edges  
segmentation\_utils, 10

app  
segmenter\_flask\_API, 19

BaseConfig, 21  
SECRET\_KEY, 21

clip\_dcm  
RadiPopGUI, 25

color\_mask  
RadiPopGUI, 25

config, 9

correct\_partition  
RadiPopGUI, 26

correctPartition  
segmenter\_flask\_API, 13

create\_image\_stream  
RadiPopGUI, 26

dcm2png  
segmenter\_flask\_API, 13

dcm2pngPreview  
segmenter\_flask\_API, 14

DEBUG  
DevelopmentConfig, 22  
TestingConfig, 35

DevelopmentConfig, 21  
DEBUG, 22

draw\_on\_image  
RadiPopGUI, 26

draw\_region\_outlines  
segmentation\_utils, 10

drawOnMask  
segmenter\_flask\_API, 14

extend\_labels  
RadiPopGUI, 27

extendLabels  
segmenter\_flask\_API, 15

extract\_mask\_and\_segmented\_flask\_API.py, 37

RadiPopGUI, 27

find\_organ  
RadiPopGUI, 27

FLASK\_PORT

segmenter\_flask\_API, 19

FLASK\_HOST  
segmenter\_flask\_API, 19

getMask  
segmenter\_flask\_API, 15

guess\_bounds  
segmentation\_utils, 10

highlight\_np\_label\_array\_to\_png  
RadiPopGUI, 29

highlightOrgan  
RadiPopGUI, 29  
segmenter\_flask\_API, 16

host  
segmenter\_flask\_API, 19

initialize  
segmenter\_flask\_API, 16

labelMask  
RadiPopGUI, 30

labelOrgan  
segmenter\_flask\_API, 17

LABELS  
RadiPopGUI, 33

last\_clicked\_x  
RadiPopGUI, 33

last\_clicked\_y  
RadiPopGUI, 33

LIVER\_LABEL  
RadiPopGUI, 33

masks  
RadiPopGUI, 34

np\_label\_array\_to\_png  
RadiPopGUI, 30

partition\_at\_threshold  
segmentation\_utils, 11

pathToSlices  
RadiPopGUI, 34

patient\_id  
RadiPopGUI, 34

- patients
  - segmenter\_flask\_API, 19
- port
  - segmenter\_flask\_API, 19
- postPickleGetMask
  - segmenter\_flask\_API, 17
- ProductionConfig, 22
- radipop\_gui, 9
- RadiPopGUI, 22
  - \_\_init\_\_, 24
  - clip\_dcm, 25
  - color\_mask, 25
  - correct\_partition, 26
  - create\_image\_stream, 26
  - draw\_on\_image, 26
  - extend\_labels, 27
  - extract\_metadata\_from\_dcm, 27
  - find\_organs, 27
  - highlight\_np\_label\_array\_to\_png, 29
  - highlightOrgan, 29
  - labelMask, 30
  - LABELS, 33
  - last\_clicked\_x, 33
  - last\_clicked\_y, 33
  - LIVER\_LABEL, 33
  - masks, 34
  - np\_label\_array\_to\_png, 30
  - pathToSlices, 34
  - patient\_id, 34
  - read\_pickle\_mask\_to\_np\_label\_array, 31
  - readPNG, 31
  - save\_masks, 32
  - selected\_pixel\_value\_of\_label\_mask, 34
  - slice\_dim, 32
  - sliceCache, 34
  - SPLEEN\_LABEL, 34
  - update\_mask\_upon\_slider\_change, 32
  - writePillow2PNG, 33
- read\_pickle\_mask\_to\_np\_label\_array
  - RadiPopGUI, 31
- readPNG
  - RadiPopGUI, 31
- save\_masks
  - RadiPopGUI, 32
- save\_partition
  - segmentation\_utils, 12
- saveMasks
  - segmenter\_flask\_API, 18
- SECRET\_KEY
  - BaseConfig, 21
- segmentation\_utils, 9
  - add\_sobel\_edges, 10
  - draw\_region\_outlines, 10
  - guess\_bounds, 10
  - partition\_at\_threshold, 11
  - save\_partition, 12
- segmenter\_flask\_API, 12
  - app, 19
  - correctPartition, 13
  - dcm2png, 13
  - dcm2pngPreview, 14
  - drawOnMask, 14
  - extendLabels, 15
  - FLASK\_PORT, 19
  - FLAST\_HOST, 19
  - getMask, 15
  - highlightOrgan, 16
  - host, 19
  - initialize, 16
  - labelOrgan, 17
  - patients, 19
  - port, 19
  - postPickleGetMask, 17
  - saveMasks, 18
  - updateMask, 18
  - selected\_pixel\_value\_of\_label\_mask
    - RadiPopGUI, 34
  - slice\_dim
    - RadiPopGUI, 32
  - sliceCache
    - RadiPopGUI, 34
  - SPLEEN\_LABEL
    - RadiPopGUI, 34
- TestingConfig, 35
  - DEBUG, 35
- update\_mask\_upon\_slider\_change
  - RadiPopGUI, 32
- updateMask
  - segmenter\_flask\_API, 18
- writePillow2PNG
  - RadiPopGUI, 33