1. Import necessary libraries:

```
import os
import sys
import random
import numpy as np
import matplotlib.pyplot as plt

from mrcnn.config import Config
from mrcnn import model as modellib, utils
from mrcnn import visualize
```

2. Configure model settings:

```
class CocoConfig(Config):
    # Adjust settings as needed, for example:
    NAME = "coco"  # Model name
    IMAGES_PER_GPU = 2  # Images per GPU during training
    NUM_CLASSES = 81  # Number of classes (80 for COCO)
```

3. Load the model:

```
model = modellib.MaskRCNN(mode="inference", config=CocoConfig(),
model_dir="logs")
model.load_weights("mask_rcnn_coco.h5", by_name=True)  # Load pre-trained COCO
weights
```

4. Prepare an image:

```
image = utils.load_image("dog.jpg")
results = model.detect([image], verbose=1)
```

5. Visualize results:

```
r = results[0]
visualize.display_instances(image, r['rois'], r['masks'], r['class_ids'],
                             class_names, r['scores'])
```

Key Steps and Visual Explanations:

1. **Import:** Bring in required libraries for model architecture, data handling, visualization, and more.
2. **Configuration:** Set up model-specific parameters like image sizes, class names, and training settings.
3. **Loading:** Load a pre-trained Mask R-CNN model, often from the COCO dataset, for instance segmentation tasks.
4. **Image Preparation:** Load an image for segmentation and pass it through the model's detection process.
5. **Visualization:** Display the segmentation results, highlighting detected objects with bounding boxes and corresponding masks.

Additional Points:

- **Custom Dataset:** Train Mask R-CNN on your own dataset by defining custom model configuration, data loading, and training routines.
- **Applications:** Utilize Mask R-CNN for various tasks like object segmentation in images, video analysis, and medical image processing.
- **Resource Availability:** Explore online tutorials and resources for detailed guidance and code examples.