# Preparing your input image

When using an already trained model like **ResNet50**, we need to make sure that we fit the network the way it was originally trained. So if we want to use a trained model on our custom images, these images need to have the same dimensions as the one used in the original model.

The original **ResNet50 model** was trained with images of size **224x224 pixels**and a number of preprocessing operations; like the subtraction of the mean pixel value in the training set for all training images.

You will go over these preprocessing steps as you prepare this dog's (named Ivy) image into one that can be classified by **ResNet50**.

- Import `image` from `keras.preprocessing` and `preprocess_input`from `keras.applications.resnet50`.
- Load the image with the right `target_size` for your model.
- Turn it into an array with `image.img_to_array()`.
- Pre-process `img_expanded` the same way the original ResNet50 training images were processed with `preprocess_input()`.

```
# Import image and preprocess_input
from keras.preprocessing import image
from keras.applications.resnet50 import preprocess_input

# Load the image with the right target size for your model
img = image.load_img(img_path, target_size=(224, 224))

# Turn it into an array
img_array = image.img_to_array(img)

# Expand the dimensions of the image
img_expanded = np.expand_dims(img_array, axis = 0)

# Pre-process the img in the same way original images were
img_ready = preprocess_input(img_expanded)
```