Code explanation

```
package org.mozilla.iot.webthing.example;
import org.json.JSONArray;
import org.json.JSONObject;
import org.mozilla.iot.webthing.Action;
import org.mozilla.iot.webthing.Event;
import org.mozilla.iot.webthing.Property;
import org.mozilla.iot.webthing.Thing;
import org.mozilla.iot.webthing.Value;
import org.mozilla.iot.webthing.WebThingServer;
import org.mozilla.iot.webthing.errors.PropertyError;
import java.io.IOException;
import java.util.Arrays;
import java.util.UUID;
//Declaration of class
public class SingleThing {
    //Creates an object of class Thing which used to control a lamp
    public static Thing makeThing() {
        Thing thing = new Thing("urn:dev:ops:my-lamp-1234",
                                "My Lamp",
                                new JSONArray(Arrays.asList("OnOffSwitch",
                                                            "Light")),
                                "A web connected lamp");
        //Creates a json structure
        JSONObject onDescription = new JSONObject();
        //Adds entries to the json structure
        onDescription.put("@type", "OnOffProperty");
        onDescription.put("title", "On/Off");
        onDescription.put("type", "boolean");
        onDescription.put("description", "Whether the lamp is turned on");
        //Adds the json structure to thing object as a property
        thing.addProperty(new Property(thing,
                                       "on",
                                       new Value(true),
                                       onDescription));
         //Creates a json structure
        JSONObject brightnessDescription = new JSONObject();
        //Adds entries to the json structure
        brightnessDescription.put("@type", "BrightnessProperty");
        brightnessDescription.put("title", "Brightness");
        brightnessDescription.put("type", "integer");
        brightnessDescription.put("description",
                                  "The level of light from 0-100");
        brightnessDescription.put("minimum", 0);
```

Code explanation (cont)

```
        brightnessDescription.put("maximum", 100);
        brightnessDescription.put("unit", "percent");
        //Adds the json structure to thing object as a property
        thing.addProperty(new Property(thing,
                                       "brightness",
                                       new Value(50),
                                       brightnessDescription));
         //Creates a json structure
        JSONObject fadeMetadata = new JSONObject();
        JSONObject fadeInput = new JSONObject();
        JSONObject fadeProperties = new JSONObject();
        JSONObject fadeBrightness = new JSONObject();
        JSONObject fadeDuration = new JSONObject();
        //Adds entries to the json structure
        fadeMetadata.put("title", "Fade");
        fadeMetadata.put("description", "Fade the lamp to a given level");
        fadeInput.put("type", "object");
        fadeInput.put("required",
                      new JSONArray(Arrays.asList("brightness", "duration")));
        fadeBrightness.put("type", "integer");
        fadeBrightness.put("minimum", 0);
        fadeBrightness.put("maximum", 100);
        fadeBrightness.put("unit", "percent");
        fadeDuration.put("type", "integer");
        fadeDuration.put("minimum", 1);
        fadeDuration.put("unit", "milliseconds");
        fadeProperties.put("brightness", fadeBrightness);
        fadeProperties.put("duration", fadeDuration);
        fadeInput.put("properties", fadeProperties);
        fadeMetadata.put("input", fadeInput);
        //Adds the json structure to thing object as an action
        thing.addAvailableAction("fade", fadeMetadata, FadeAction.class);
        JSONObject overheatedMetadata = new JSONObject();
        overheatedMetadata.put("description",
                               "The lamp has exceeded its safe operating temperature");
        overheatedMetadata.put("type", "number");
        overheatedMetadata.put("unit", "degree celsius");
        thing.addAvailableEvent("overheated", overheatedMetadata);
        //returns the thing object with the added project
        return thing;
    }
     //Startpoint for the program
    public static void main(String[] args) {
```

Code explanation (cont)

```
        //Creates an object of thing and the needed properties are added
        Thing thing = makeThing();
        WebThingServer server;
        try {
            // If adding more than one thing, use MultipleThings() with a name.
            // In the single thing case, the thing's name will be broadcast.
            server = new WebThingServer(new WebThingServer.SingleThing(thing),
                                        8888);
            //Hook is added so the program can be shut down
            Runtime.getRuntime().addShutdownHook(new Thread() {
                public void run() {
                    server.stop();
                }
            });
            //Starts the server and if something unexpected happens it prints the error and then exits
              the program
            server.start(false);
        } catch (IOException e) {
            System.out.println(e);
            System.exit(1);
        }
    }
```

Comments are in code

Packet Tracer

1) Set up the wired network with connecting FashEthernet0 on PC to Home Switch
2) Set up the wireless network with DHCP on the Home Gateway
3) Connect IoT devices to the network; Connect the Coffee Pot with FastEthernet0 by DHCP to the existing network
4) To access IoT remotely it needs a server address, user name and password.
5) IoT can be accessed remotely with a web browser, the address, user name and password