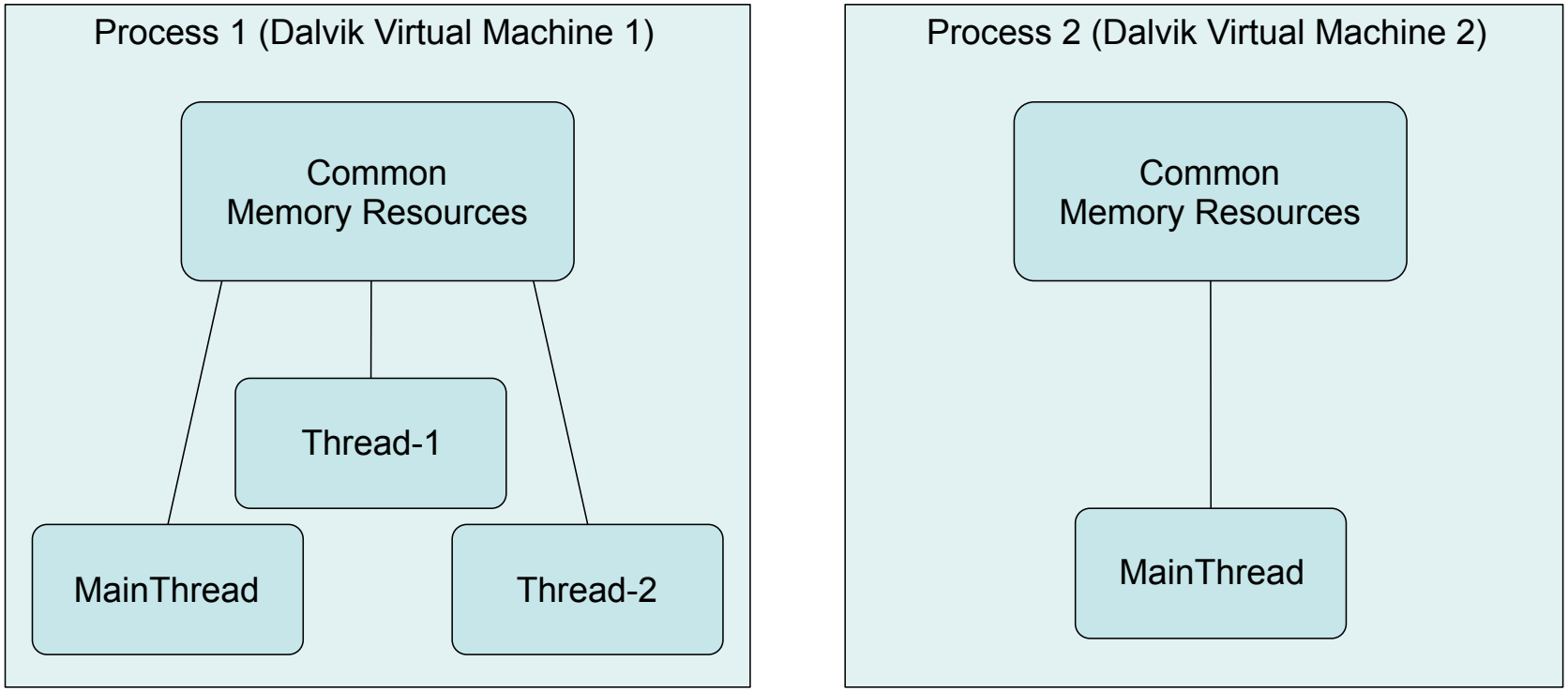
Processes and threads

- A Thread is a unit of concurrent execution
- A single process can have multiple threads that share global data and address space with other threads running in the same process



Java threads

- Building blocks for threads in Java:
  - An class that implements `Runnable` interface. All execution takes place in the `run()` method of an object of this type.
    - (One often provides the needed references and parameters in a constructor)
  - A `Thread` object to which the runnable object is attached (again, often in `Thread` constructor)
- Some methods of `Thread` class instances:
  - `start()`, `sleep()`, `join()`
  - (possibly inside a `try-catch` block)

Java threads

- Start the execution of the `run()` method of the `Runnable` object attached to a thread by calling `start()` method of the thread
  - Calling `start()` is asynchronous - the call returns immediately
  - `run()` method is executed in parallel with the thread that called `start()`
  - After `run()` finishes the thread is dead, it is not possible to restart it
- If needed, it is possible to wait for a thread to complete its execution by calling `join()`

AsyncTask – asynchronous operation without an explicit thread

- Allows background operations and results publishing on the UI thread without having to manipulate threads and/or handlers.
- You can specify the type, using generics, of the *parameters*, the *progress values* and the *final value* of the task
- The method `doInBackground()` executes automatically on a worker thread - you don't need to create thread yourself
- `onPreExecute()`, `onPostExecute()` and `onProgressUpdate()` are all invoked on the UI thread - they can perform operations on the state of UI
- The value returned by `doInBackground()` is sent to `onPostExecute()`
- You can call `publishProgress()` at anytime in `doInBackground()` to execute `onProgressUpdate()` on the UI thread
- You can cancel the task at any time, from any thread

AsyncTask

```
public class DownCounter extends AsyncTask<Integer, Float, String> implements … {
     …

    public DownCounter(…) {
        …
    }

    @Override
    protected String doInBackground(Integer... params) {
        if(params.length < 2) {
            return "No success.";
        }

        for(…) {
            …
            publishProgress(… a float expression …);
        }

        return "Done.";
    }

    @Override
    protected void onProgressUpdate(Float... percentage) {
        …
    }

    @Override
    protected void onPostExecute(String s) {
        …
   }
}
```

These are run in UI thread - can call activity methods that update UI state. Adapt (simplified) observer pattern to make it possible.

Getting started with networking

- Declare in manifest your intentions to do networking

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

- Create URL for addressing your request and connect

```
URL url = new URL("http://10.0.2.2:8080/WebApplication4/webresources/Players");
HttpURLConnection conn = (HttpURLConnection)url.openConnection();
```

- Modify parameters as needed (defaults are good for reading, ie. doing a GET)

```
conn.setDoOutput(true);
conn.setDoInput(true);
conn.setRequestMethod("POST");
conn.setRequestProperty("Content-Type", "application/xml; charset=utf-8");
```

… and do the real thing

- Use needed Java IO class(es) for doing IO

```
Scanner sc = new Scanner(conn.getInputStream());

while(sc.hasNextLine()) {
    int id = Integer.parseInt(sc.nextLine());
    …
}

sc.close();
```

- Disconnect

```
conn.disconnect();
```

- (… and pay attention to return code)

```
conn.getResponseCode()
```

Reading list

- Processes and threads:
  - http://developer.android.com/guide/components/processes-and-threads.html
  - http://developer.android.com/reference/android/os/AsyncTask.html
- Networking:
  - http://developer.android.com/training/basics/network-ops/connecting.html
  - (http://developer.android.com/training/basics/network-ops/managing.html)