

Example07

NOTE

Continuo dell'esempio [Example06](#) con soluzione al problema di sincronizzazione visto alla fine.

Table of contents

- [`it.unipr.informatica.concurrent`](#)
 1. [`Callback.java`](#)
 2. [`ExecutorService.java`](#)
 1. [Implementazioni di `Callback.java`](#)
- [Example07](#)
 1. [`DownloadManager.java`](#)
 2. [`Example07.java`](#)

it.unipr.informatica.concurrent

Callback.java

In altri contesti, proprio a causa della sincronizzazione molto forte imposta dai `Future`, per far sì che i task ritornino valori, non si utilizzano i `Future`, vengono usati i `Callback`. In `java.util.concurrent` non c'è, la creiamo noi.

Usiamo il thread che esegue il `Callable` anche per processare il risultato, viene *tutto eseguito insieme*.

```
package it.unipr.informatica.concurrent;

public interface Callback<T> {
    public void onSuccess(T result);
    public void onFailure(Throwable throwable);
}
```

ExecutorService.java

Aggiungiamo i metodi molto simili agli altri, ma che accettano `Callback` come argomento.

```
package it.unipr.informatica.concurrent;

public interface ExecutorService extends Executor {
    // ...

    public void submit(Runnable task, Callback<?> callback);
    public <T> void submit(Callable<T> task, Callback<T> callback);
}
```

Implementazioni di Callback.java

```
// ...
public void submit(Runnable task, Callback<?> callback) {
    if(task == null)
        throw new NullPointerException("task == null");
    if(callback == null)
        throw new NullPointerException("callback == null");
}
```

```

        execute(() → {
            try {
                task.run();
                callback.onSuccess(null);
            } catch (Throwable throwable) {
                callback.onFailure(throwable);
            }
        });
    }
    // ...

```

```

// ...
public <T> void submit(Callable<T> task, Callback<T> callback) {
    if(task == null)
        throw new NullPointerException("task == null");
    if(callback == null)
        throw new NullPointerException("callback == null");
    execute(() → {
        try {
            T result = task.call();
            callback.onSuccess(result);
        } catch (Throwable throwable) {
            callback.onFailure(throwable);
        }
    });
}
// ...

```

Example07

DownloadManager.java

Il metodo `download` non ritornera' piu' un `Future`, ma prendera' una `Callback`.

```

// ...
public void download(String url,
    Callback<ResourceContent> callback) {
    if(url==null)
        throw new IllegalArgumentException("url==null");
    executorService.submit(() →
        downloadResourceContent(url), callback);
}
// ...

```

Example07.java

```

package it.unipr.informatica.examples;

public class Example07 {
    private void process(ResourceContent content) {
        System.out.println("Downloaded " + content.getData().length +
            " bytes from " + content.getURL());
    }

    private void go() {
        DownloadManager downloadManager = new DownloadManager(4);
        downloadManager.download("https://www.google.it",
            this::process);
        downloadManager.download("https://www.youtube.it",

```

```
        this::process);
downloadManager.download("https://www.amazon.it",
        this::process);
downloadManager.download("https://www.missingwebsite.com",
        this::process);
downloadManager.download("https://www.ebay.it",
        this::process);
downloadManager.download("https://www.unipr.it",
        this::process);
downloadManager.shutdown();
}

public static void main(String[] args) {
    new Example07().go();
}
}
```