# Rich Web Application Technology

## **Exercises**

**Q1. Explain what is meant by the stream abstraction. What is the relationship between**

**streams and the observer pattern? What are streams useful for modelling and when**

**might you use them in Rich Web development?**

Streams is an abstract concept because it can be applied in many different situations; in other words, it is a general concept. For example, in java 8 streams can be implemented via the Stream API or asynchronous events in JavaScript. Streams involve transporting data in an incremental fashion i.e. the data is sent in chunks which are typically a single byte in size. Streams are similar to arrays in that they offer similar operations which can be invoked on the object. They may be implemented when you are unsure of the potential size of the data that will arrive into the application; you can access the data straight away.

Streams implement the design pattern called the observer pattern hence there is a relationship between the two. The observer pattern allows streams to work with asynchronous functionalities through the use of observables. The concept of the observer pattern involves a list of observers which are subscribed to a subject which in turn identifies an event. . Each time there is a change, all the observers are notified. In Java this is implemented by using listeners whereas in JavaScript it uses event handlers.

When we have data that lives in various remote locations it introduces a synchronisation problem. One such method of overcoming this problem is to use streams to model the application state. If all aspects of the application are using streams then the application is now a stream processing problem rather than a synchronisation problem. Streams are used for reactive programming. Reactive programming utilises asynchronous events to handle updating of the application.

**Q2. Assume that you are building an interface to an API in your Rich Web App. Describe in detail how you could use the RxJS library to handle asynchronous network responses to API requests. In your opinion, what are the benefits to using a streams library for**

**networking over, say, promises? And what do you think are the downsides?**

RxJS can be used to handle network responses to API requests by implementing an observable stream which observers can subscribe to. For example, when interacting with an API using RxJS we can utilise various operators such as map, flat map and tap etc. If we wanted to retrieve some information from an API we would use observables which are a core concept in RxJS. These observables contain data as it becomes available. An observer will be able to see the data coming into an observable. When interacting with an API we could use the Fetch API to make a HTTP request which will return an observable which can perform HTTP requests when subscribed to by an observer.

Streams provide many more functionalities than promises. Streams can handle multiple asynchronous events over a given period of time whereas a promise is limited to one asynchronous event over a given time. In addition, promises cannot be cancelled whereas as observables do allow cancelling by invoking unsubscribe. Streams provide array like operators whereas promises do not provide any sort of operations. One downside to streams is that they are more complex to implement, such that if we were developing a simple application implementing streams would be overkill and unnecessary whereas promises would be more appropriate.

**Q3. Consider three asynchronous tasks, A,B & C. What are the consequences of these**

**functions sharing global state? What is a good practice to alleviate any problems**

**associated with this?**

If all three asynchronous tasks use the same global state many problems will be introduced. For example if tasks A and B are both running simultaneously and task A alters the state of a global variable and is dependent on this change further on in the task but then task B changes the state of the global variable before task A could use it then we will run into a problem. Another issue presented that the code will be harder to debug if all three tasks are sharing the same space and altering the same variables etc.

Good practice is to either avoid using global variables altogether or the actual value of the global variable could be passed to the task via function parameters(pass by value).