**Deliverable #8**

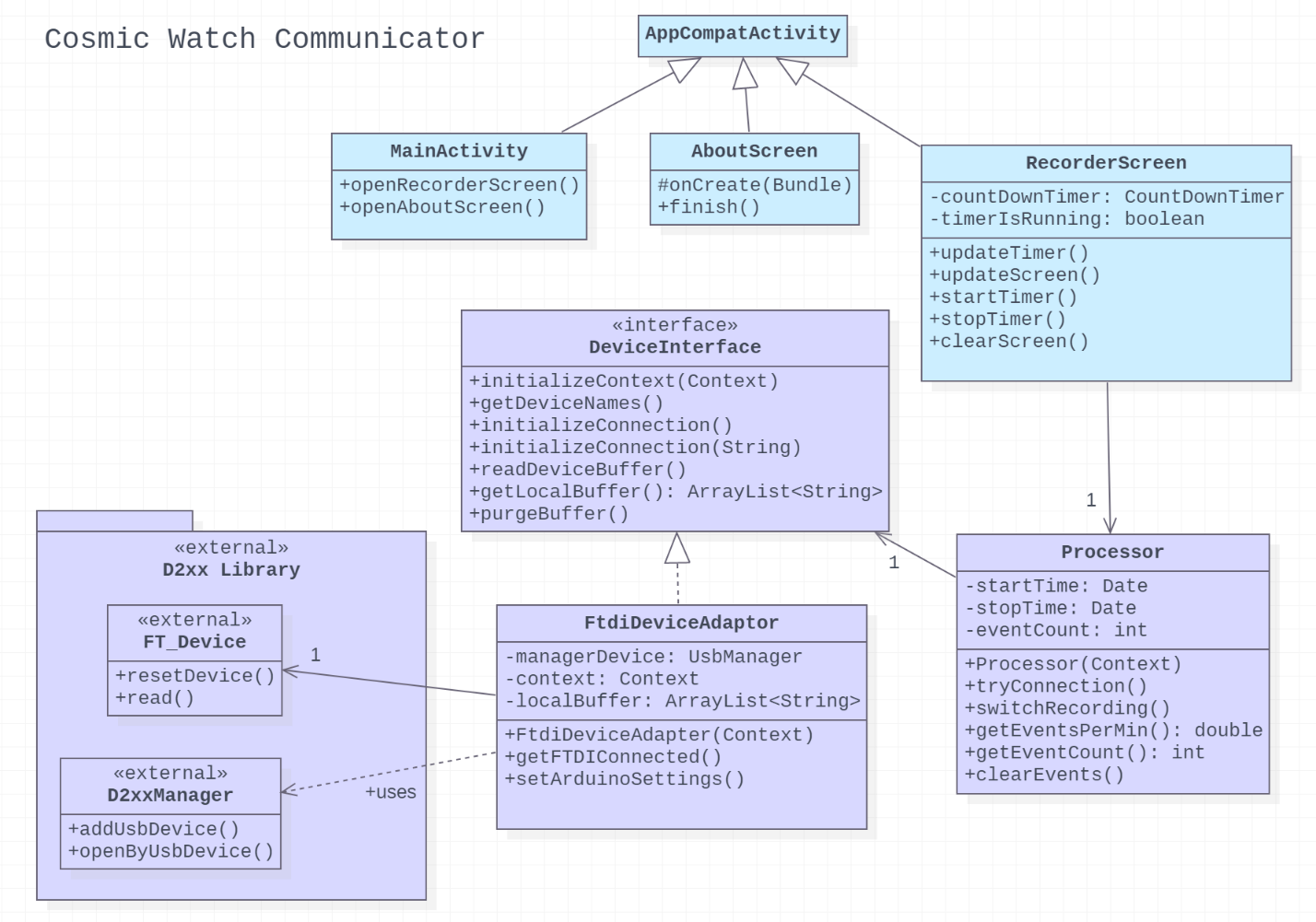


Figure xx: A class diagram for our application. Classes that are part of the Adapter design pattern are shown in purple.

We decided to implement the Adapter design pattern in our project. In our implementation, the Processor class acts as a client. The target interface is the DeviceInterface class, which contains methods used by the Processor to communicate with the muon detector. The FtdiDeviceAdaptor implements this interface to translate requests from the Processor to the third-party D2xx Library. This library is the adaptee which directly handles connecting to the muon detector.

This pattern was chosen to make our product more flexible in future updates. As our application develops, it may need to support reading data from devices that are not compatible with the external D2xx library. With our updated code, we can now use adapters implementing DeviceInterface to handle specific cases.

With this design pattern, the Processor only uses methods defined by the DeviceInterface to receive data from an external device, regardless of the type of device it is. This reduces coupling since the Processor is only associated with the interface, and not individual adapters. The cohesion of classes has also been increased, since each adapter added will handle a particular type of device, rather than having one class managing many at once.