Adapter Pattern

|  |  |
| --- | --- |
| Intent: | Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn’t otherwise because of incompatible interfaces. |
| Structure: | The pattern may realize its form in the shape of either a class adapter or object adapter type implementation. The pattern should use whichever implementation style is better suited to ensuring the characteristics desired of the system in which is utilized. If the class adapter pattern is chosen, multiple inheritance will be required, so its correct-functioning may be limited in certain cases or in systems where the language support for multiple-inheritance is non-existent (such as java). If the object adapter pattern is chosen, object composition is used, so the adapter can adapt a particular class that it knows of and any of that classes subclasses, as well. (As my implementation is in java, my demo uses the object adapter type implementation, and it suites my needs fine) |
| Applicability: | 1. You want to use an existing class, and its interface does not match the one you need; (my implementation capitalizes on this) 2. You want to create a reusable class that cooperates with unrelated or unforeseen classes; that is, classes that don’t necessarily have compatible interfaces. (my implementation capitalizes on this) |
| Consequences: | Object Adapter implementation makes it harder to override Adaptee behavior and requires additional sub-classing and making the Adapter subclass refer to it – which may not be possible if we do not have the code of the Adapter class in the beginning. |
| Brief Description: | My demo uses FileChooserDemo to run the program (aka 'Client'). It presents a pop-up file chooser, from which 'Sources.xml' (in the .zip'ed directory) can be selected. The program will attempt to convert Sources.xml - an MSO-formatted bibliography file - into a format which BibTex can understand displayed in the console.  Process described in steps:  1) FileChooserDemo creates a FormatAdapter object and Adaptee object.  - Adaptee object's constructor requires a File as its parameter.. The file is an XML file containing Raw bibliography data which is parsed(/adapted?) into a familiar Java list-structure for easier manipulation.  2) The adapt() method of the Adaptee Object (of type MSOtoBIBconverter which implements Adaptee's interface) is used to do the actual conversion, so it is the actual delegate to which the Adapter (FormatAdapter) assigns implementation responsibility.  3) Client (FileChooserDemo) can access the Target-friendly formatted-bibliography (BibTex Format) for any Adaptee-types that are valid and have been associated to FormatAdapter as an Adaptee.  Disclaimer: the assignment does not require fulfilling the feature but understanding the pattern, the program is not fully functional for producing truly bibTex-compatible output from the converter type supplied – this is just a matter of completely refining the converter class. |

Future Research: ACM describes their publication formats as [essentially dependent upon LaTex](http://www.acm.org/publications/latex_style/) in their [Author Guidelines](http://cacm.acm.org/about-communications/author-center/author-guidelines). Mendeley provides an exciting opportunity for interested developers/academic researchers to more easily manager their references (freely) and is exploring the role of scientific literature recommendations in more an analytics project at: [dev.mendeley.com/datachallenge/](http://dev.mendeley.com/datachallenge/)