Parsing XML with lxml

The XML above shows two appointments. The beginning time is in seconds since the epoch; the uid is generated based on a hash of the beginning time and a key; the alarm time is the number of seconds since the epoch, but should be less than the beginning time; and the state is whether or not the appointment has been snoozed, dismissed or not. The rest of the XML is pretty self-explanatory. Now let's see how to parse it.

```
from lxml import etree

def parseXML(xmlFile):
    """
    Parse the xml
    """
    with open(xmlFile) as fobj:
        xml = fobj.read()

    root = etree.fromstring(xml)

    for appt in root.getchildren():
        for elem in appt.getchildren():
            if not elem.text:
                text = "None"
            else:
                text = elem.text
            print(elem.tag + " => " + text)

if __name__ == "__main__":
    parseXML("example.xml")
```

First off, we import the needed modules, namely the **etree** module from the lxml package and the **StringIO** function from the built-in **StringIO** module. Our **parseXML** function accepts one argument: the path to the XML file in question. We open the file, read it and close it. Now comes the fun part! We use etree's parse function to parse the XML code that is returned from the StringIO module. For reasons I don't completely understand, the parse function requires a file-like object.

Anyway, next we iterate over the context (i.e. the **lxml.etree.iterparse object**) and extract the tag elements. We add the conditional if statement to replace the empty fields with the word “None” to make the output a little clearer. And that’s it.