Hashtables

Hashtables (also called dictionaries, maps, or associative arrays) are data structures that allow you to store named items. They are similar to Python's dictionaries. For example:

```
d = {"foo" : "bar", "spam" : "eggs"}

print d["foo"]  //prints "bar"

emptydict = {}

print d.GetType() //-> Boo.Lang.Hash
```

Boo's hash class is defined in Boo.Lang.Hash, which is a subclass of the standard System.Collections.Hashtable cla ss. See that page for more documentation about the methods and properties that boo's hashtable inherits.

If you haven't already, you should learn about Lists And Arrays also.

Here is a sample of things you can do with a dictionary:

```
d = {"foo" : "bar", "spam" : "eggs"}

print "d has $(d.Count) items"

//add a new item:
d["test"] = "new item"

print "d has ${d.Count} items"

//change an existing item
d["foo"] = "barbar"

//print everything in the dictionary
for item in d:  //item is of type
```

```
System.Collections.DictionaryEntry
 print item.Key, ":", item.Value

//an alternative way
for key in d.Keys:
 print key, "->", d[key]

if d.ContainsKey("foo"):
 print "has key foo"

if d.ContainsValue("eggs"):
 print "has value eggs"

//get the first key: (have to explicitly
convert to an array)
print "first key:", array(d.Keys)[0]

//print the first value
print "first value:", array(d.Values)[0]

//convert hash to a jagged array like
python's dictionary.items:
items = array((item.Key, item.Value) for
item in d)
//(('spam', 'eggs'), ('foo', 'barbar'),
('test', 'new item'))


//Remove an item:
d.Remove("test")
```

//Getting a default value if the key is not found:

```
//This works because d["badkey"] will return
null
item = d["badkey"] or "default value"
```

See also Lists And Arrays.