WARNING! This method doesn't work. Well, it works actually but disables all other shortcuts, so don't do it. I'm currently working on finding a working method. I'll keep you posted.

This post is a followup of [this blog post](http://dennis.dieploegers.de/handling-keyboard-shortcuts-in-zimbra-zimlets/). The old post was based on Zimbra 6.x. We're at 8.0.4 by the time of this writing and Zimbra has changed and evolved.

The methods of handling keyboard shortcuts in Zimbra has been improved since that and I can strike out some of the things I didn't understand in the original version. So this is the way to enable your zimlet to react on shortcuts based on Zimbra 8.x (and probably 7.x):

To be able to handle shortcuts you still have to create a class, that inherits DwtKeyMap:

```
de_dieploegers_exampleKeyMap = function () {
DwtKeyMap.call(this);
this._load(
this._map,
de_dieploegers_example
);
};
```

de_dieploegers_exampleKeyMap.prototype = new DwtKeyMap(true);
de_dieploegers_exampleKeyMap.prototype.constructor = de_dieploegers_exampleKeyMap;

But that's all. You don't have to create a key map and create a crude reversed map to achieve very simple shortcut handler. Please note the parameter „de_dieploegers_example" parameter to the „_load" method. That is the name of the object, that is automatically generated by Zimbra when interpreting the internationalization properties files.

If you fire up a web development environment (like the Chrome developer tools) when inside the Zimbra development mode((Add a ?dev=1 to your Zimbra URL to get there)) you'll see a javascript object named after your zimlet. So if you have named your zimlet „de_dieploegers_example" in your zimlet definition file, that is the name of the object that holds the translated strings and that is the parameter you have to specify in the „_load"-method.

Now, add these lines to your zimlet handler object's init-method:

```
var kbMgr;
```

kbMgr = appCtxt.getKeyboardMgr();

kbMgr.registerKeyMap(new ca_uoguelph_ccs_archiveKeyMap());
kbMgr.pushDefaultHandler(this);

The line „kbMgr.pushDefaultHandler(this);" tells the keyboard manager to rely on your zimlet for key handling. In theory, you can create another class, that does all the key handling, but for basic tasks, this is sufficient. But this forces you to add two methods to your zimlet (my handler object is called „de_dieploegers_exampleHandlerObject"):

```
de_dieploegers_exampleHandlerObject.prototype.getKeyMapName = function () {
```

return "SHORTCUTS";

};

de_dieploegers_exampleHandlerObject.prototype.handleKeyAction = function (
actionCode,
ev
) {

// do the actual key handling

};

Now, here's something that has changed drastically: You have to tell the keyboard manager a context key map name for your shortcuts and it looks inside your properties files for this context.

Like before, the actual key assignment is done using the internationalization features of Zimbra, which still is pretty cool. So to add an actual key map you have to put these lines in your properties file:

```
SHORTCUTS.SHOW_MESSAGE.description = Show example message
SHORTCUTS.SHOW_MESSAGE.display = Shift+A
SHORTCUTS.SHOW_MESSAGE.keycode = Shift+65
```

To generalize: a property key for a shortcut assignment now looks like this:

..

The parameter can be:

* description: A description for the key mapping (currently only used by Zimbra internally for the hotkey help page)
* display: The user-friendly display of the key combination (currently only used by Zimbra internally for the hotkey help page)
* keycode: The actual keycode consisting of modifier-keywords (Shift, Alt, Ctrl, Meta) and an ASCII-based key number

Additionally, you can even filter out a platform by appending „mac“, „win“ or „linux“ to the key, separated by a dot. (i.e. „SHORTCUTS.SHOW_MESSAGE.display.mac = Cmd+A“)

And that is all. Your handler method is called by the keyboard manager with the action code assigned to the pressed shortcut (in this case „SHOW_MESSAGE“, when you pressed the Shift-key and the „a“-key) and you’re good to go.