CS403: Programming Languages

Notes on Adding Built-in Functions

Simple, but ugly

Here is a simple way to add built-in functions to your language. It's ugly, but it works. For example, let's add a *println* built-in. One does so by adding some tests to the *evalCall* function. Originally, this function looks like:

```
function evalCall(t,env)
    {
    //this code assumes a function call of the form f(x,y)
    var name = getCallName(t);
    var args = getCallArgs(t);
    var eargs = evalArgs(args,env);
    var closure = lookup(env,name);
    var params = getClosureParams(closure);
    var body = getClosureBody(closure);
    var senv = getClosureEnvironment(closure);
    var xenv = EnvExtend(senv,params,eargs);

    return eval(body,xenv);
    }
```

We modify the function to check for a call to the built-in function:

```
function evalCall(t,env)
    {
    //this code assumes a function call of the form f(x,y)
    var name = getCallName(t);
    var args = getCallArgs(t);
    var eargs = evalArgs(args,env);
    //check for built-in functions here
    if (stringEquals(name,"println"))
        return evalPrintln(eargs);
    else
        {
        var closure = lookup(name,env);
        var params = getClosureParams(closure);
        var body = getClosureBody(closure);
        var senv = getClosureEnvironment(closure);
        var xenv = EnvExtend(senv,params,eargs);

        return eval(body,xenv);
        }
    }
```

and dispatch to the appropriate handler for the built-in. Finally, add the *evalPrintln* function:

```
function evalPrintln(eargs)
    {
    while (eargs != null)
        {
        display(eargs.left);
        eargs = eargs.right;
        }
    }
```

You can perform similar actions for each of your built-ins.

A better way

To be written