SLACK BOTS & SCOPE

TO TEST YOUR BOT LOCALLY

1. *At the command line, navigate to the main folder for your bot.*
2. *Type* `./bin/hubot` *and press* **Enter**. *This gives a few warning messages that you can safely ignore, as long as the BASH prompt is not displayed again after them.*
3. *Press* **Enter**, *after which you should see a prompt such as* `hubot>` *that lets you interact with your bot.*
4. *When you're done testing, press* **control + C** *to return to the BASH prompt.*

TO TEST YOUR BOT ON SLACK

1. *At the command line, navigate to the main folder for your bot.*
2. *Push your code to Heroku using the following commands:*
   ```
   git add .
   git commit -m "description of changes"
   git push heroku master
   ```
3. *Open a direct message with the bot you're sharing in the class Slack organization (such as redbot or greenbot).*
4. *Send a direct message to interact with your bot.*
   Remember that the bot's replies may reflect the code of the classmate sharing your bot on Slack, as well as your own code, so you may get multiple replies to a single test message.

VAR, LET, & CONST

keyword	local scope	mutable	browsers
`var`	**function** only	yes	all
`let`	any block	yes	**modern**
`const`	any block	**no**	**modern**

FUNCTIONS & HOISTING

type	name hoisted	content hoisted
declaration	yes	**yes**
`let` expression	**no**	no
`var` expression	yes	no

GLOBAL, LOCAL, & BLOCK SCOPE

a variable declared outside of a function is in the **global scope**

```
let temp = 75;
```

```
function predict() {
  let forecast = 'Sun';
  console.log(temp + ' and ' + forecast);
  // 75 and Sun
}
```

a variable declared within a function is in the **local scope** of that function

```
if (temp > 70) {
  let forecast = 'It's gonna be warm!';
  console.log(temp + '! ' + forecast);
  // 75! It's gonna be warm!
}
```

a variable declared with `let` and within a block, such as an `if` statement, is in the **block scope** of that statement