What is the Shell

The kernel and the Utilities
The Login Shell
Typing Commands to the Shell
The Shell’s Responsibilities
The Kernel and the Utilities

- The Unix system is itself logically divided into two pieces: the kernel and the utilities.
The Utilities and the Shell

• Virtually every command under the Unix system is classified as a utility.
• The utility program resides on the disk and is brought into memory only when you request that the command be executed.
• The Shell is a utility program. It is loaded into memory for execution whenever you log in to the system.
The Login Shell

• A terminal is connected to a Unix system through a direct wire, modem, or network.

• For each physical terminal port on a system, the Unix system program init automatically starts up a getty program (login:) which then starts up a login program(password:).
The Login Shell

• After correct login, login then checks for the name of a program to execute. In most cases, this will be the shell program. In other cases, it may be a special custom-designed program.

• User can set up a login account to automatically run any program whatsoever whenever someone logs in to it. The shell program just happens to be the program most often selected.
The Login Shell

• The init program starts up other programs similar to getty for networked connections. For example, sshd, telnetd, and rlogind are started to service logins via ssh, telnet, and rlogin, respectively.

• These programs connect user’s shells to *pseudo ttys* that are devices emulating terminals over network connections.
Typing Commands to the Shell

• Command cycle
Typing Commands to the Shell

- Login cycle
The Shell

• It’s important to recognize that the shell is just a program.
• It has no special privileges on the system, meaning that anyone with the capability and devotion can create his own shell program.
• This is in fact the reason why various flavors of the shell exist today, including the older Bourne shell, developed by Stephen Bourne; the Korn shell, by David Korn; the “Bourne again shell,” mainly used on Linux systems; and the C shell, by Bill Joy.
The Shell’s Responsibilities

- Program Execution
- Variable and Filename Substitution
- I/O Redirection
- Pipeline Hookup
- Environment Control
- Interpreted Programming Language
Program Execution

• The Shell is responsible for all programs that you request from your terminal.

  `program-name argument`

• The line that is typed to the shell is known more formally as the command line.

• White space characters: space character, horizontal tab character, and the end-of-line characters (known more formally as the newline character)
Program Execution

• Example:

```plaintext
mv tmp/mazewars games
```

echo when do we eat?
Program Execution

There are some commands that the shell knows how to execute itself. These are built-in commands including `cd`, `pwd`, and `echo`. 
Variable and Filename Substitution

• The shell lets you assign values to variables. Whenever you specify one of these variables on the command line, preceded by a dollar sign, the shell substitutes the value assigned to the variable at that point.

• The shell also performs filename substitution on the command line. (*, ?, or [...]). For example, echo *.
I/O Redirection

• It scans the command line for the occurrence of the special redirection characters <, >, or >>.

Examples

echo Remember to tape Law and Order > reminder
$ wc -l users

$ wc -l < users
Pipeline Hookup

• The shell scans the command line looking for the pipe character |.

• For each such character that it finds, it connects the standard output from the command preceding the | to the standard input of the one following the |. It then initiates execution of both programs.

  for example, who | wc -l
Environment Control

• The shell provides certain commands that let you customize your environment:
  – Home directory
  – The characters that the shell displays to prompt you to type in a command
  – A list of the directories to be searched whenever you request that a program be executed.
Interpreted Programming Language

• The shell has its own built-in programming language.
• This language is interpreted, meaning that the shell analyzes each statement in the language one line at a time and then executes it.
• The shell programming language features: looping constructs, decision-making statements, variables, and functions, and it procedure-oriented.
• Modern shells based on the IEEE POSIX standard have features including arrays, data typing, and built-in arithmetic operations.