

# Getting Keyboard Input

---

## The Strategy

Here is a very basic strategy for getting keyboard input from the user:

1. Halt the program.
2. Wait for the user to type some characters.
3. Turn those characters into a data type Java can understand.
4. Store the input in a variable.



For example, if you want to have the user enter something you can interpret as an `int`, you have to stop, wait for the user to type characters, and then turn those characters into a number in the `int` format.

Fortunately, the `hsafx` Console contains special methods that take care of all that.

## Getting Characters with `c.getChar()`

You already know about one input method: `c.getChar()`. You know it halts and waits for the user to press a key. If you want to know what key they pressed, store the result in a variable of type `char`.

1. Create a variable to store the character.
2. Prompt the user (a prompt is a message, like "Please press a key.")
3. Call `c.getChar()` and store the result in the variable you created.

Here's what it looks like when you put it all together

```
char favLetter;  
c.println("What's your favorite letter?");  
favLetter = c.getChar();  
c.println("You chose " + favLetter + "!");
```

## Getting Integers with `c.nextInt()`

If you want the user to enter an integer, you can use `c.nextInt()`, like this:

```
int age;  
c.println("Please enter your age.");  
age = c.nextInt();  
c.println("You entered: " + age);
```

The `hsafx` library also contains `c.nextDouble()` for getting doubles, and `c.nextLine()` for getting Strings.

## A New Kind of Error: Run-Time Errors

There are three types of errors possible in any program. You are probably already familiar with the first two, but user input makes a third kind of error possible.

### Syntax Errors

These are **errors that prevent compilation** (for example, a missing semicolon or extra bracket).

### Logic Errors

The program compiles and runs without crashing, but it **doesn't do what you expected**.

### Run-Time Errors

These are **errors that crash the program while it is running**. User input can cause run-time errors in a program (for example, you want an integer, but the user enters letters). There are other ways to cause them as well (for example, division by zero). In hsafx, run-time errors halt the program and cause an error message to pop up.

## Four Basic Input Methods

There is a different method for every variable type. Here are three for four types you already know about.

### **c.nextInt ()**

Waits for the user to type and hit enter, then returns the number the user typed as an **int**. Run-time error if the user types something other than an `int` (like "6.7" or "4g5")

### **c.nextDouble ()**

Waits for the user to type and hit enter, then returns the number the user typed as a **double**. Run-time error if something other than a `double` (like "4g5" or "7.8.9")

### **c.nextLine()**

Waits for the user to type and hit enter, then returns the entire line as a **String**.

### **c.getChar()**

Waits for the user to press a button and then returns the key pressed as a **char**.

