

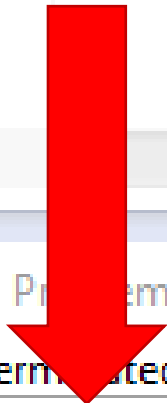
# Exceptions

Error handling

```
AcceptMe.java
```

```
public class AcceptMe {  
    public static void main(String args[]) {  
        new AcceptMe();  
    }  
    public AcceptMe(){  
        String one = "hello";  
        char two = one.charAt(88);  
    }  
}
```

AcceptMe  
S main(S)  
C Accept



Problems @ Javadoc Declaration Console

<terminated> AcceptMe [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (Sep 14, 2017 7:18:41 AM)

Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 88  
at java.lang.String.charAt(Unknown Source)  
at AcceptMe.<init>(AcceptMe.java:8)  
at AcceptMe.main(AcceptMe.java:4)

```
AcceptMe.java
```

```
public class AcceptMe {  
    public static void main(String args[]) {  
        new AcceptMe();  
    }  
    public AcceptMe(){  
        String one = "hello";  
        int i = Integer.parseInt(one);  
    }  
}
```

AcceptMe  
main(S  
Accept

Problems @ Javadoc Declaration Console

<terminated> AcceptMe [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (Sep 14, 2017 7:21:41 AM)  
Exception in thread "main" java.lang.NumberFormatException: For input string: "hello"  
 at java.lang.NumberFormatException.forInputString(Unknown Source)  
 at java.lang.Integer.parseInt(Unknown Source)  
 at java.lang.Integer.parseInt(Unknown Source)  
 at AcceptMe.<init>(AcceptMe.java:8)  
 at AcceptMe.main(AcceptMe.java:4)

AcceptMe.java

```
public class AcceptMe {  
    public static void main(String args[]) {  
        new AcceptMe();  
    }  
    public AcceptMe(){  
        String one;  
        String two = one.substring(0, 3);  
    }  
}
```

AcceptMe  
S main(S)  
C Accept

Problems @ Javadoc Declaration Console

<terminated> AcceptMe [Java Application] C:\Program Files (x86)\Java\jre7\bin\javaw.exe (Sep 14, 2017 7:23:20 AM)

Exception in thread "main" java.lang.Error: Unresolved compilation problem:  
The local variable one may not have been initialized

at AcceptMe.<init>(AcceptMe.java:8)  
at AcceptMe.main(AcceptMe.java:4)

AcceptMe.java

```
public class AcceptMe {  
    public static void main(String args[]) {  
        new AcceptMe();  
    }  
    public AcceptMe(){  
        try {  
            String one = "hello";  
            char two = one.charAt(88);  
        }  
        catch (Exception e) {  
            System.out.println("Be careful");  
        }  
    }  
}
```

Problems @ Javadoc Declaration Console

<terminated> AcceptMe [Java Application] C:\Program Files (x86)\Java\jre7\bin\java  
Be careful

AcceptMe.java

```
public class AcceptMe {  
    public static void main(String args[]) {  
        new AcceptMe();  
    }  
    public AcceptMe(){  
        try {  
            String one = "hello";  
            int i = Integer.parseInt(one);  
        }  
        catch (NumberFormatException e){  
            System.out.println("That's not a number!");  
        }  
        catch (Exception e) {  
            System.out.println("Be careful");  
        }  
    }  
}
```

Problems @ Javadoc Declaration Console

<terminated> AcceptMe [Java Application] C:\Program Files (x86)\Java\jre7\bin'

That's not a number!

# Try/Catch Blocks

- They are used to handle exceptions (errors) so that your code doesn't just crash on the user.
- They are like an if:
  - If the code has no error at run-time, use the try block.
  - Otherwise, if it has an error, use the catch block.
- Files are especially error prone, so Java requires try/catch for them.
  - The file might be renamed
  - The file might be moved
  - The file might be deleted

```
try{
```



Exceptions....  
got to catch 'em all.

```
}catch( Exception ){  
    //Do nothing  
}
```



# Exception-Handling

try...



catch...



```
while ( strlen("LIFE") == 2 * sizeof(short)) {
```

```
    try {
```

```
        followYourDreams();
```

```
    } catch (exception &failure) {
```

```
        learnFromIt();
```

```
    }
```

```
}
```

if you never  
try {}  
you'll never  
catch ()

**I TRIED TO  
CATCH SOME FOG,  
BUT I MIST**

There is no `try {}`

(Maybe that's why I'm `Exceptional`)

## Why use Try/Catch Blocks: Separating Error-Handling Code from "Regular" Code

```
readFile {  
    open the file;  
    determine its size;  
    allocate that much memory;  
    read the file into memory;  
    close the file;  
}
```

## Advantage 1: Separating Error-Handling

What happens if the file can't be opened?

Code

```
readFile {  
    open the file;  
    determine its size;  
    allocate that much memory;  
    read the file into memory;  
    close the file;  
}
```

## Advantage 1: Separating Error-Handling

What happens if the file can't be opened?

"Error" Code

What happens if the length of the file can't be determined?

```
readFile {  
    open the file;  
    determine its size;  
    allocate that much memory;  
    read the file into memory;  
    close the file;  
}
```



## Advantage 1: Separating Error-Handling

```
readFile {  
    open the file;  
    determine its size;  
    allocate that much memory;  
    read the file into memory;  
    close the file;  
}
```

What happens if the file can't be opened?

What happens if the length of the file can't

What happens if enough memory can't be allocated??

"Error" Code

## Advantage 1: Separating Error-Handling

```
readFile {  
  open the file;  
  determine its size;  
  allocate that much memory;  
  read the file into memory,  
  close the file;  
}
```

What happens if the file can't be opened?

"Error" Code

What happens if the length of the file can't

What happens if enough memory can't

What happens if the read fails?

## Advantage 1: Separating Error-Handling

```
readFile {  
  open the file;  
  determine its size;  
  allocate that much memory;  
  read the file into memory,  
  close the file;  
}
```

What happens if the file can't be opened?

What happens if the length of the file can't

What happens if enough memory can't

What happens if the read fails?

What happens if the file can't be closed?


"Error" Code

```
errorCodeType readFile {
    initialize errorCode = 0;

    open the file;
    if (theFileIsOpen) {
        determine the length of the file;
        if (gotTheFileLength) {
            allocate that much memory;
            if (gotEnoughMemory) {
                read the file into memory;
                if (readFailed) {
                    errorCode = -1;
                }
            } else {
                errorCode = -2;
            }
        } else {
            errorCode = -3;
        }
        close the file;
        if (theFileDidntClose && errorCode == 0) {
            errorCode = -4;
        } else {
            errorCode = errorCode and -4;
        }
    } else {
        errorCode = -5;
    }
    return errorCode;
}
```



Yuck.  
Nested  
ifs.



Ahh.  
Clean  
Code.

```
readFile {
    try {
        open the file;
        determine its size;
        allocate that much memory;
        read the file into memory;
        close the file;
    } catch (fileOpenFailed) {
        doSomething;
    } catch (sizeDeterminationFailed) {
        doSomething;
    } catch (memoryAllocationFailed) {
        doSomething;
    } catch (readFailed) {
        doSomething;
    } catch (fileCloseFailed) {
        doSomething;
    }
}
```