Class java.lang.Throwable

```
java.lang.Object
   |
   +----java.lang.Throwable
```

---

public class **Throwable**
extends Object

An object signalling that an exceptional condition has occurred. All exceptions are a subclass of Exception. An exception contains a snapshot of the execution stack, this snapshot is used to print a stack backtrace. An exception also contains a message string. Here is an example of how to catch an exception:

```
try {
    int a[] = new int[2];
    a[4];
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("an exception occurred: " + e.getMessage());
    e.printStackTrace();
}
```

Version:
1.22, 10/17/95

Constructor Index

o **Throwable**()
Constructs a new Throwable with no detail message.
o **Throwable**(String)
Constructs a new Throwable with the specified detail message.

Method Index

o **fillInStackTrace**()
Fills in the excecution stack trace.
o **getMessage**()
Gets the detail message of the Throwable.
o **printStackTrace**()

Prints the Throwable and the Throwable's stack trace.

- **toString**()
  Returns a short description of the Throwable.

Constructors

- **Throwable**

```
public Throwable()
```

Constructs a new Throwable with no detail message. The stack trace is automatically filled in.

- **Throwable**

```
public Throwable(String message)
```

Constructs a new Throwable with the specified detail message. The stack trace is automatically filled in.

Parameters:

message – the detailed message

Methods

- **getMessage**

```
public String getMessage()
```

Gets the detail message of the Throwable. A detail message is a String that describes the Throwable that has taken place.

Returns:

the detail message of the throwable.

- **toString**

```
public String toString()
```

Returns a short description of the Throwable.

Overrides:

toString in class Object

- **printStackTrace**

```
public void printStackTrace()
```

Prints the Throwable and the Throwable's stack trace.

- **fillInStackTrace**

```
public Throwable fillInStackTrace()
```

Fills in the excecution stack trace. This is useful only when rethrowing a Throwable. For example:

```
try {
     a = b / c;
} catch(ArithmeticThrowable e) {
     a = Number.MAX_VALUE;
     throw e.fillInStackTrace();
}
```

Returns:

the Throwable itself.

See Also:

printStackTrace