{.js}

JavaScript

Exception Handling

A R U N   A R U N I S T O

Exception handling in JavaScript allows developers to handle unexpected errors or exceptional situations in their code. JavaScript provides **try**, **catch**, and **finally** blocks to handle exceptions.

Try-Catch Statement:

The **try** block allows you to define a block of code to be tested for errors while it is being executed. If an error occurs within the **try** block, it throws an exception, and the control goes to the **catch** block.

Syntax:

```
try {
  // code block
} catch (error) {
  // code block when error occurs
}
```

In the **catch** block, you can handle the **exception** and take appropriate actions. The **error** object contains information about the exception, such as the **error message**.

Example:

```
try {
  let result = undefinedVariable/0;
  console.log(result);
} catch (error) {
  console.log("And the error is:",error.message);
}
```

Finally Block:

The **finally** block allows you to execute code, regardless of whether an exception is thrown or caught. The code inside the **finally** block will always be executed, regardless of whether an exception occurred in the **try** block or not.

Syntax:

```
try {
  // code block
} catch (error) {
  // code block when error occurs
} finally {
  // always run if error occur/not
}
```

Example:

```
try {
  let result = 9/19;
  console.log("result:",result);
} catch (error) {
  console.log("Error:",error.message);
} finally {
  console.log("Finally Code");
}
```

Custom Errors:

In JavaScript, you can create and throw custom errors using the throw keyword. This allows you to handle specific error cases in your code by creating custom error objects. Custom errors can be instances of the built-in Error object or objects that inherit from Error.

Syntax:

```
throw new Error(<error_name>)
```

Example:

```
function divide(a, b) {
  if (b === 0) {
    throw new Error("Zero Division Error");
  } else if ((typeof(a) === 'string') || (typeof(b) ===
'string')) {
    throw new Error("value must be integer");
  }
  else {
    return a/b;
  }
}

try {
  let total = divide(19, 0);
  console.log(total);
} catch (e) {
  console.log(e);
}
```