Validating Data and RegEx

Download PDF Copy

Validating Data

Preventing Exceptions

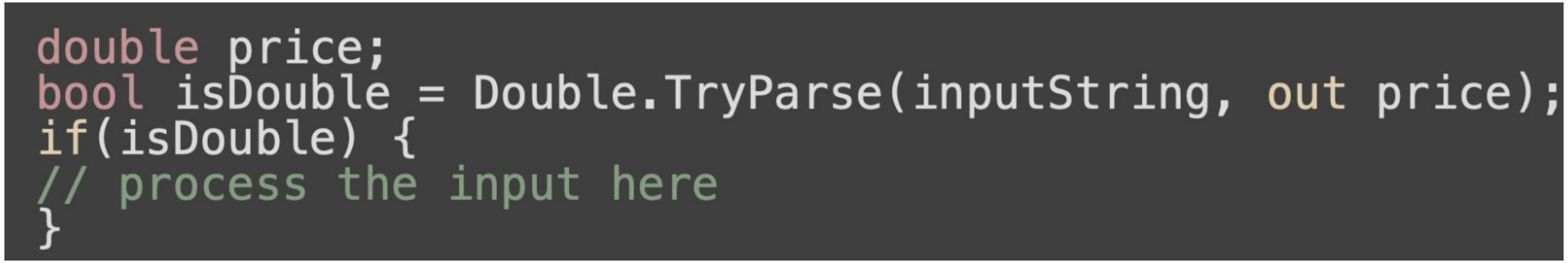
We will generally wish to prevent exceptions being thrown due to invalid data. We can use data validation to display an error message when the user inputs invalid content until all entries are valid.

Numeric Entries

- Make sure the entry has a valid numeric format
- Make sure that the entry is within a valid range. This is known as range checking.

Validating String Input

Use the TryParse!



```
double price;
bool isDouble = Double.TryParse(inputString, out price);
if(isDouble) {
// process the input here
}
```

Using Regular Expressions To Validate Input

Regular Expressions are used for pattern matching, where you don't look for exact matches.

- Example: Find if the input is a valid email!

Using Regular Expressions To Validate Input

A pattern consists of one or more character literals, different structures, or operators.

- Example: \d{5} : looks for a match with 5 consecutive digits.

Examples On Regular Expression Patterns

- \d : single digit
- [a-z] : small case char between a and z.
- [abc] : a or b or c
- \d*: 0 or more digits
- \d+: 1 or more digits
- \d{3} : exactly 3 digits
- ab|cd : ab or cd

Examples On Regular Expression Patterns

- You can find more patterns on https://regexr.com/
- You can use this site to build and test out regular expressions you build.
- Example: Let us build a regular expression to match a password that can have at least 3 letters (can be numbers or alphabets).

Examples On Regular Expression Patterns

```
if (Regex.IsMatch("555-1212", "^\d\d\d-\d\d\d\d$"))
{
// If the IsMatch method returns true, that means that you have a valid input!
}
```