Characters: The std.uni Module

In this lesson, we explore the functions of the std.uni module.

WE'LL COVER THE FOLLOWING

The `std.uni` module

The `std.uni` module includes functions that are useful with unicode characters. You can see this module in its documentation.
The functions that start with "**is**" answer certain questions about characters. The result is false or true depending on whether the answer is no or yes, respectively. These functions are useful in logical expressions:

- `isLower`: is it a lowercase character?
- `isUpper`: is it an uppercase character?
- `isAlpha`: is it a Unicode alphabetic character?
- `isWhite`: is it a whitespace character?

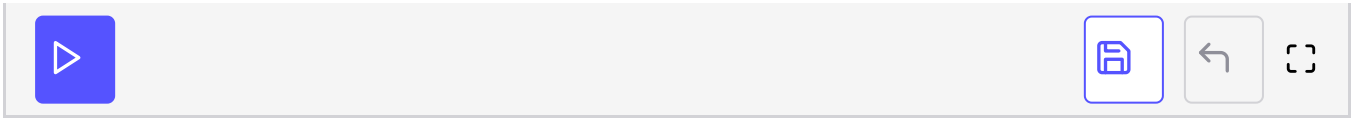
The functions that start with "**to**" produce new characters from existing ones:

- `toLower`: produces the lowercase version of the given character.
- `toUpper`: produces the uppercase version of the given character.

Here is a program that uses all these functions:

```
import std.stdio;
import std.uni;

void main() {
```

```
    writeln("Is g lowercase? ", isLower('g'));
    writeln("Is $ lowercase? ", isLower('$'));


    writeln("Is İ uppercase? ", isUpper('İ'));
    writeln("Is ç uppercase? ", isUpper('ç'));

    writeln("Is z alphabetical? ",       isAlpha('z'));
    writeln("Is \&euro; alphanumeric? ", isAlpha('\&euro;'));

    writeln("Is new-line whitespace? ",  isWhite('\n'));
    writeln("Is underscope whitespace? ", isWhite('_'));

    writeln("The lowercase of Ğ: ", toLower('Ğ'));
    writeln("The lowercase of İ: ", toLower('İ'));

    writeln("The uppercase of ş: ", toUpper('ş'));
    writeln("The uppercase of ı: ", toUpper('ı'));
}
```



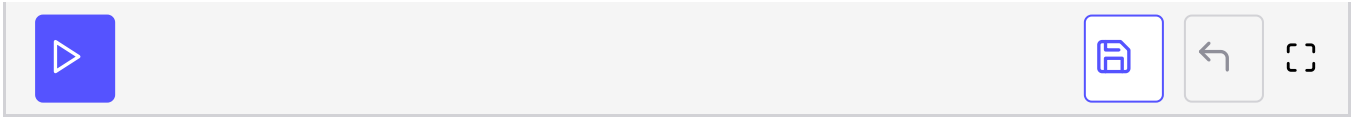
std.uni module functions

Limited support for ı and i

The lowercase and uppercase versions of the letters ‘ı’ and ‘i’ are consistently dotted or undotted in some alphabets (e.g., the Turkish alphabet). Most other alphabets are inconsistent in this regard: the uppercase of the dotted ‘i’ is undotted ‘I’.

Because the computer systems started with the ASCII table, traditionally, the uppercase of ‘i’ is ‘I’ and the lowercase of ‘I’ is ‘i’. For that reason, these two letters may need special attention. The following program demonstrates this problem:

```
import std.stdio;
import std.uni;

void main() {
    writeln("The uppercase of i: ", toUpper('i'));
    writeln("The lowercase of I: ", toLower('I'));
}
```



i and I limited support

The output is according to the basic Latin alphabet:

```
The uppercase of i: I
The lowercase of I: i
```

Characters are converted between their uppercase and lowercase versions normally by their Unicode character codes. This method is problematic for many alphabets. For example, the Azeri and Celt language alphabets are subject to the same problem of producing the lowercase of 'I' as 'i'.
There are similar problems with sorting: Many letters like 'ğ' and 'á' may be sorted after 'z' even for the basic Latin alphabet.

In the next lesson, we will see problems with reading characters and D's Unicode support.