UTF-32 support

Unicode character support of ES6 and comparison between the for-of and for-in loops

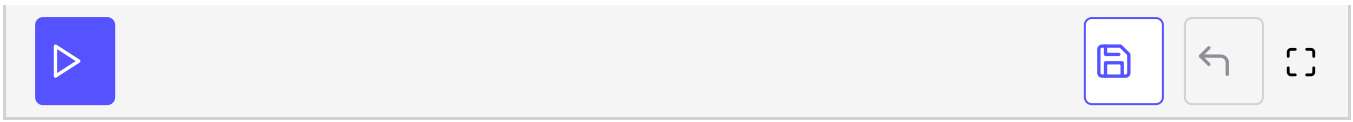
Let me give you some foreshadowing about the next section, where we will deal with template literals. You will learn about the Unicode character support of ES6. You will also learn that in ES5, three and four byte long characters are sometimes broken into 2-byte chunks.

```
let text = '\u{1F601}\u{1F43C}';
console.log( 'text: ', text );

for( let i in text ) {
    console.log(text[i]);
};

console.log('-----');

for ( let c of text ) {
    console.log( c );
};
```



Execute the above code.

The for-of loop handles UTF-32 characters properly. The for-in loop interprets them as two byte long characters.

> Unlike the for-in loop, the for-of loop parses all Unicode characters properly, regardless of their size.

Now, let's move on to destructuring using the for-of loop.