# NETHACK PROJECT JS

<https://www.nethack.org/>

## ABSTRACT CLASSES

<https://stackoverflow.com/questions/29480569/does-ecmascript-6-have-a-convention-for-abstract-classes>

## DOCUMENT.WRITE()

- Write JS to the html document

<https://www.w3schools.com/jsref/met_doc_write.asp>

## CLASS

- Function associated with class is a method

- Object created from class is an instance

### PRIVATE VARIABLES

- Use ‘\_’ on the variable

- Then place method in the constructor to access

- Access with obj.getSymbol()

**class** Dog **extends** Tile{

**constructor**(symbol){

super();

**let** \_symbol = symbol;

this.getSymbol=()**=>**{

return \_symbol;

}

}

}

### INHERITENCE

- Avoids duplication of code

- Data and methods are passed from Tile to Dog

**class** Tile{

**constructor**(symbol,isPassable){

this.symbol = symbol; *// This will represent the tile*

this.isPassable = isPassable; *// This will determin if can pass this tile*

this.getSymbol = () **=>** {

*// Fetch symbol to be displayed*

return this.symbol;

}

if (new.target === Tile) {

*// Will act like abstract class, where cant make class from this*

throw new TypeError("Cannot construct Abstract instances directly");

}

}

}

**class** Dog **extends** Tile{

**constructor**(symbol,isPassable){

super(symbol, isPassable); *// Inherits from tile class (superclass)*

}

}

#### IS-A AND LIKE-A

<https://en.wikipedia.org/wiki/Liskov_substitution_principle>

- Inheritance should always model an IS-A and WORKS-LIKE-A relationship. That is, a manager “is a” and “works like a” specific kind of employee,

#### HAS-A

- If inheritance models the IS-A relationship, then composition models the HAS-A relationship

- Where a class can contain a type of another class, but isnt a is-a or a like-a relationship

## IMPORT ONE CLASS TO ANOTHER USING NODE.JS

- Need to export the class

module.exports = **class** Tile{}

- Then to require the class where you want to use it

**var** Tile = require('./Tile.js');

## MODIFY INHERITED METHOD

<https://javascript.info/class-inheritance>

doAttack(){

super.doAttack();

}

## CHECK IF CONTENT IS SUBCLASS OF ANOTHER

<https://stackoverflow.com/questions/18939192/how-to-test-if-b-is-a-subclass-of-a-in-javascript-node>

if(val.getContent() instanceof Item){

console.log("hit " + val.getContent().getEffect());

}

## GETTING FILE FROM INPUT

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/file>

<https://developer.mozilla.org/en-US/docs/Web/API/File/Using_files_from_web_applications>

- Select the first file in the nodes file list, returns File Object

var file = document.getElementById('fileItem').files[0];

<https://developer.mozilla.org/en-US/docs/Web/API/FileReader/readAsText>

### THIS WORKS!!!

<https://stackoverflow.com/questions/5201317/read-the-contents-of-a-file-object>

- When loop through the text file, the \n shows up as an enter arrow

<https://stackoverflow.com/questions/35808205/javascript-replace-from-string>

*class* Gameboard{

*constructor*(file){

*// Takes filename as string*

*this*.*file* *=* file; *// 2D array of Tiles*

*this*.*board*;

}

*toString=*()*=>*{

*let* boardContent *= {*

'#'*: new Wall(),*

'.'*: new OpenTile()*

*}*

*var* reader *= new FileReader()*;

*// reader.readAsText(this.board);*

*reader*.*readAsBinaryString*(*this*.*file*);

*reader*.*onloadend* *=* *function* () {

*// reader.result returns string with board*

*// Save the resulting string*

*let* boardString *= reader.result*;

*// Split result into array*

*let* boardArray *= boardString.split(*''*)*;

*let* rowArray *= []*;

*this*.*board* *=* []; *// Set constructor variable to empty array*

*for*(*let* i *= 0*; i*<boardArray*.length;i*++*){

*if* (boardArray[i] *===* '\n' *&&* boardArray[i*+*1]*!==*'\n'){

*// If newline then dont add this to the array*

*// Also if multiple newlines in a row then dont add them*

*let* array *= [...*rowArray*]*

*// Only after first line push the array*

*this*.*board*.*push*(array);

rowArray *=* []; *// Erase array*

}

*else* *if*(boardArray[i]*!==*'\n'){

*// Push the content into the array if not a newline*

*rowArray*.*push*(boardContent[boardArray[i]]);

}

};

console.log(*this*.*board*);

}

*// console.log(boardRead+" text")*

*// Method that will print the board*

*// Put current health at the top*

*// console.log(this.board)*

}

}

*// Select the first file in the nodes file list, returns File Object*

*const inputElement = document.getElementById(*"fileItem"*)*;

*inputElement*.*addEventListener*("change", (e)*=>*{

*var* file *= document.getElementById(*'fileItem'*).files[0]*;

*// const fileList = this.files; /\* now you can work with the file list \*/*

*let* gameboard *= new Gameboard(*file*)*;

*gameboard*.*toString*();

})