

untitled

```
1 /**
2  * Class => Weapon(name, damage)
3  * -----
4  * Creates a weapon item.
5  * Weapon items can be equipped for use in battle.
6  *
7  * The Weapon class constructor will call
8  *   the super class (Item) constructor
9  *   while passing in the 1 Item constructor param
10 *
11 * @name Weapon
12 * @param {string} name      The weapon's name.
13 * @param {number} damage    The weapon's damage.
14 * @property {number} damage
15 */
16
17
18
19 /**
20  * Weapon Extends Item Class
21  * -----
22 */
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
```

untitled

```
1  /**
2   * Class => Weapon(name, damage)
3   * -----
4   * Creates a weapon item.
5   * Weapon items can be equipped for use in battle.
6   *
7   * The Weapon class constructor will call
8   * the super class (Item) constructor
9   * while passing in the 1 Item constructor param
10  *
11  * @name Weapon
12  * @param {string} name      The weapon's name.
13  * @param {number} damage    The weapon's damage.
14  * @property {number} damage
15  */
16
17 function Weapon () {
18 }
19
20 /**
21  * Weapon Extends Item Class
22  * -----
23  */
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
```

Class Name
@name



```
untitled

1 /**
2  * Class => Weapon(name, damage)
3  * -----
4  * Creates a weapon item.
5  * Weapon items can be equipped for use in battle.
6  *
7  * The Weapon class constructor will call
8  * the super class (Item) constructor
9  * while passing in the 1 Item constructor param
10 *
11 * @name Weapon
12 * @param {string} name      The weapon's name.
13 * @param {number} damage    The weapon's damage.
14 * @property {number} damage
15 */
16
17 function Weapon (name, damage) {
18 }
19
20 /**
21  * Weapon Extends Item Class
22  * -----
23  */
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
```

The diagram illustrates the mapping of JSDoc annotations to the code. It shows three annotations in the documentation block:

- * @param {string} name
- * @param {number} damage
- * @property {number} damage

Arrows point from these annotations to the corresponding parameters in the constructor:

- An arrow from the first annotation points to the 'name' parameter in the constructor.
- An arrow from the second annotation points to the 'damage' parameter in the constructor.
- An arrow from the third annotation points to the 'damage' property in the constructor.

A large arrow points from the constructor definition to the label "Parameters @param".

untitled

```
1  /**
2  * Class => Weapon(name, damage)
3  * -----
4  * Creates a weapon item.
5  * Weapon items can be equipped for use in battle.
6  *
7  * The Weapon class constructor will call
8  *   the super class (Item) constructor
9  *   while passing in the 1 Item constructor param
10 *
11 * @name Weapon
12 * @param {string} name      The weapon's name.
13 * @param {number} damage    The weapon's damage.
14 * @property {number} damage
15 */
16
17 function Weapon (name, damage) {
18
19     this.damage = damage;
20 }
21
22
23 /**
24 * Weapon Extends Item Class
25 * -----
26 */
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
```

Property
@property

Property Assignment

Parameters
@param



untitled

```
1  /**
2  * Class => Weapon(name, damage)
3  * -----
4  * Creates a weapon item.
5  * Weapon items can be equipped for use in battle.
6  *
7  * The Weapon class constructor will call
8  * the super class (Item) constructor
9  * while passing in the 1 Item constructor param
10 *
11 * @name Weapon
12 * @param {string} name      The weapon's name.
13 * @param {number} damage    The weapon's damage.
14 * @property {number} damage
15 */
16
17 function Weapon (name, damage) {
18     this.damage = damage;
19     Item.call(this, name);
20 }
21
22 /**
23 * Weapon Extends Item Class
24 * -----
25 */
26
27
28
29
30
31
32 invoke super class constructor method
33
34
35
36
37
38
39
40
41
```

Super Class => Item

Super class constructor Parameter
@param

Context => Weapon Class

```
1  /**
2   * Class => Weapon(name, damage)
3   * -----
4   * Creates a weapon item.
5   * Weapon items can be equipped for use in battle.
6   *
7   * The Weapon class constructor will call
8   * the super class (Item) constructor
9   * while passing in the 1 Item constructor param
10  *
11  * @name Weapon
12  * @param {string} name      The weapon's name.
13  * @param {number} damage    The weapon's damage.
14  * @property {number} damage
15  */
16
17 function Weapon (name, damage) {
18
19     this.damage = damage;
20
21     Item.call(this, name);
22 }
23
24 /**
25  * Weapon Extends Item Class
26  * -----
27  */
28
29 Weapon.prototype = Object.create(Item.prototype);
```

Sub Class

Super Class

Sub Class extends Super Class prototype
Sub Class inherits all Super Class methods

untitled

```
1  /**
2   * Class => Weapon(name, damage)
3   * -----
4   * Creates a weapon item.
5   * Weapon items can be equipped for use in battle.
6   *
7   * The Weapon class constructor will call
8   *   the super class (Item) constructor
9   *   while passing in the 1 Item constructor param
10  *
11  * @name Weapon
12  * @param {string} name      The weapon's name.
13  * @param {number} damage    The weapon's damage.
14  * @property {number} damage
15  */
16
17 function Weapon (name, damage) {
18
19     this.damage = damage;
20
21     Item.call(this, name);
22 }
23
24 /**
25  * Weapon Extends Item Class
26  * -----
27  */
28
29 Weapon.prototype = Object.create(Item.prototype, {
30     constructor: {
31         value: Item
32     }
33 });
34
35     Inherits the Super Class
36     constructor method
37
38
39
40
41
```