Extra assignment III

You have just been hired as a senior programmer at a new startup called **FlagShip!** and your job is to create a new product called **Mothership Ultra** and you want to show that you are a hot stuff developer and know your game. While working on this product you discover that the native **Object** in JavaScript just isn‘t providing you with all the functionality you need. You decide to extend **Object.prototype** and show the new guys that you know your stuff when it comes to JavaScript and objects.

Assignment description

Extend the **Object.prototype** with the following functionality:

- (**10%**) Create an extension method called **merge(firstObject, secondObject, ..., nObject)** which should combine all objects in to a single object. If there are properties which have the same name the first object should have the deciding factor. Returns a single merged object

- (**10%**) Create an extension method called **pick(prop, …, propN)** which should be executed on an instance of **Object** and should return a new object with only the properties stated as parameters. If the property is not found, ignore it.

- (**10%**) Create an extension method called **tail()** which returns the last property of the object. If no property resides within the **Object**, undefined should be returned.

- (**10%**) Create an extension method called **head()** which returns the first property of the object. If no property resides within the **Object**, undefined should be returned.

- (**10%**) Create an extension method called **remove(predicateFunction)** which takes in a function which should be used to determine what to remove from the object.

- (**10%**) Create an extension method called **difference(firstObject, secondObject)** which takes in two objects and returns the difference of these objects as a new object *(possibly an empty object)*

- (**10%**) Create an extension method called **intersection(firstObject, firstObject)** which takes in two objects and returns the intersection of these objects as an object *(possibly an empty* object)

- (**10%**) Create an extension method called **hasIn(elem)** which takes in a property and returns true or false whether the property exists or not.

- (**10%**) Create an extension method called **updatePath(path : string, updater : func)** which takes in a path and a function called updater. The path should determine where the property resides within the object, e.g. { x: { y: 1 } } can have a path to the *y* property with ‘x.y’. By calling { x: { y: 1 } }.updatePath(‘x.y’, (n) => n * 2), the *y* property should have changed to **2**

- (**10%**) Create an extension method called **count()** which should count have many properties reside within the **Object**

Submission
Submit a single .js file containing your **Object.prototype** extension methods.

Resources
You may not use any additional libraries to implement this solution.