Summary

- Function invocations rely upon implicit execution context that resolves to the global object
- Method invocations rely upon implicit context that resolves to the object that holds the method
- All JS code executes within a context. The top level context in a web browser is the `window` object
    - All global methods and Objects (`NaN` or `Math`) are properties of this object
- You can't use `delete` to delete variables and functions declared at the global scope
- `this` is the current execution context of a function
- The value of `this` changes based on how you invoke a function, not how you define it
- JS has first-class functions which have following characteristics
    - You can add them to objects and execute them in the respective object's contexts
    - You can remove them from their objects, pass them around, and execute them in different contexts
    - They're initially unbound but dynamically bound to a context object at execution time
- `call` and `apply` invoke a function with an explicit execution context
- `bind` permanently binds a function to a context and returns a new context
- Method invocatinos can operate on the data of the owning object