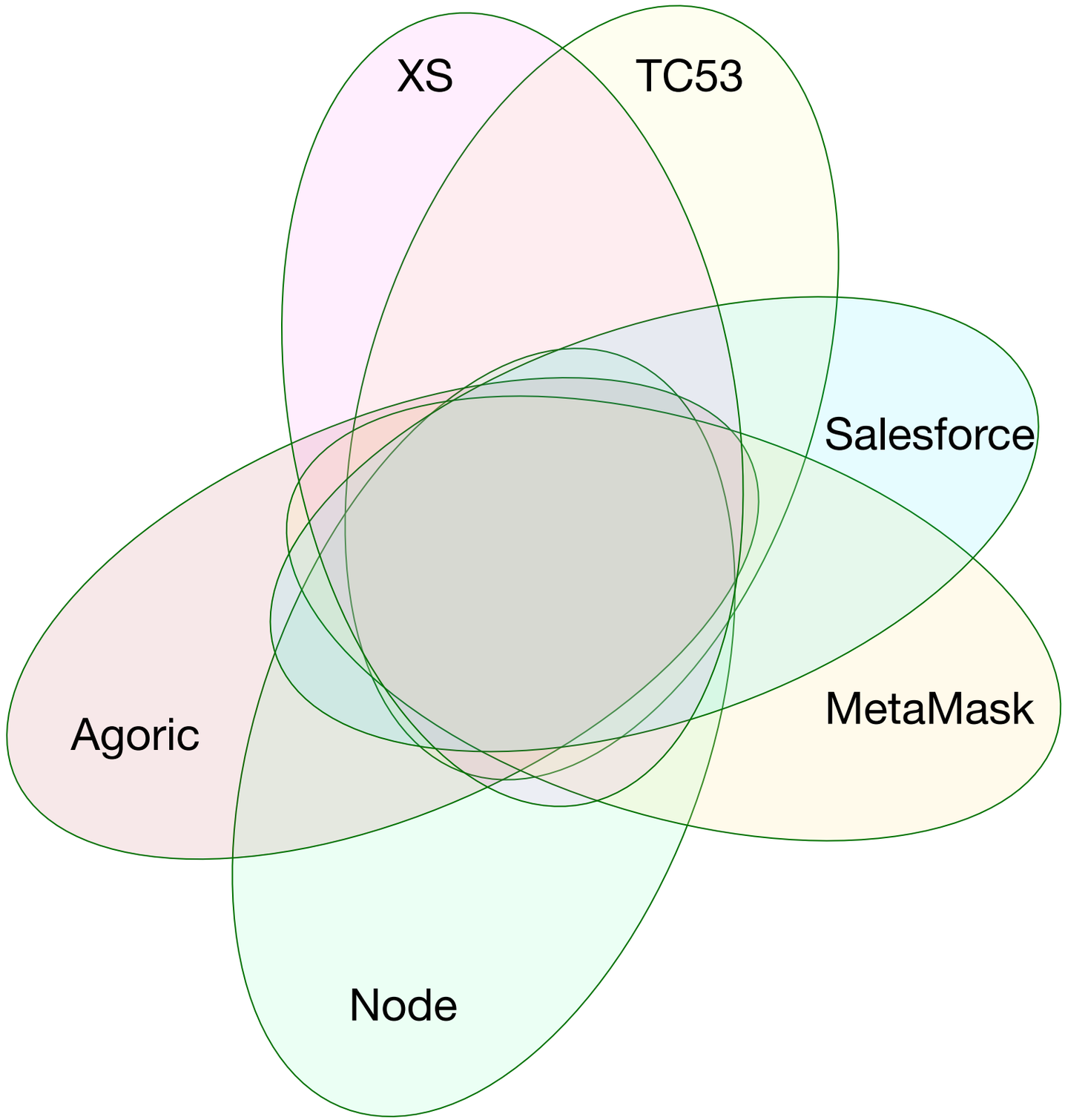
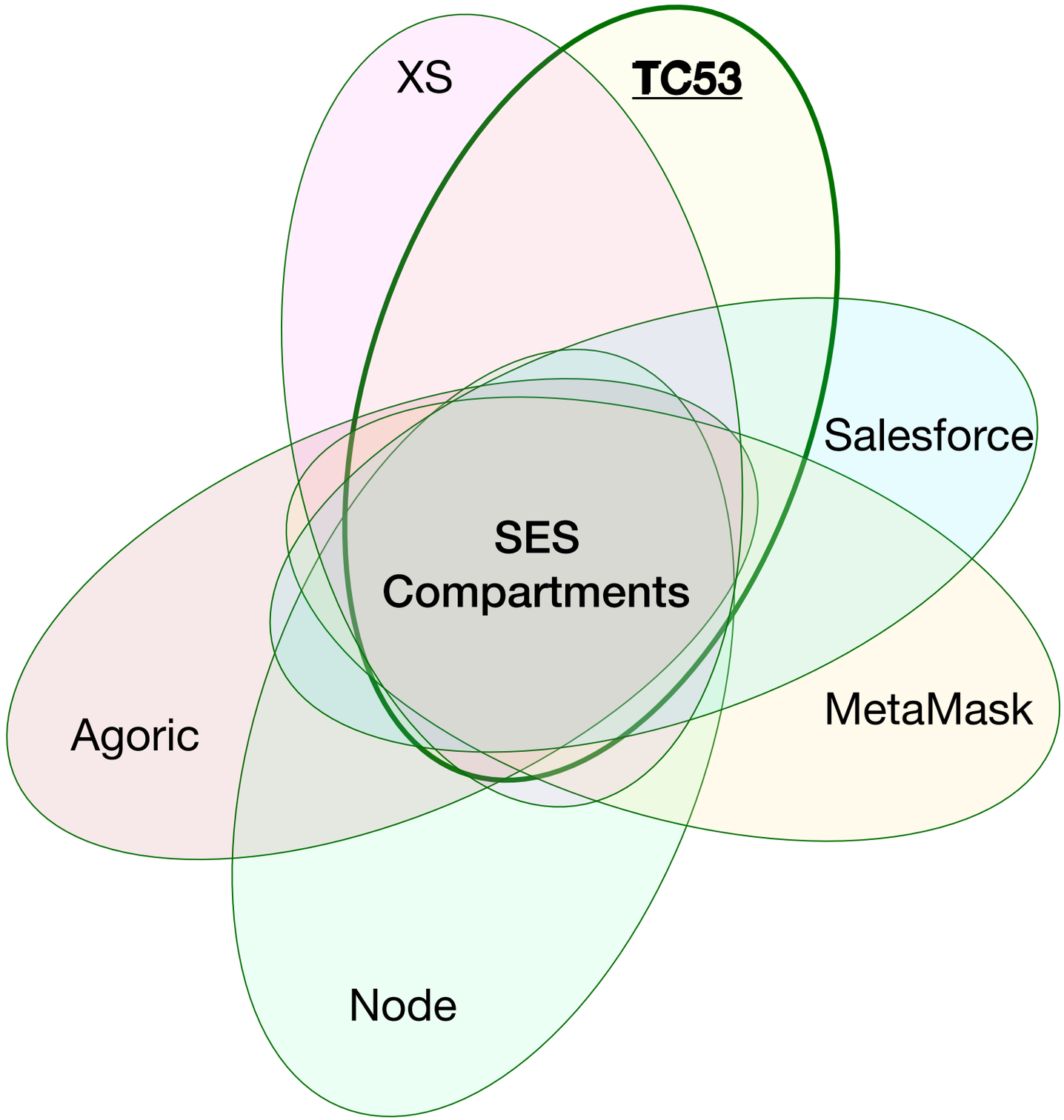
SES Compartments

Mark S. Miller, Agoric
JF Paradis, Agoric
Caridy Patiño, Salesforce
Patrick Soquet, Moddable
Bradley Farias, GoDaddy, Node

tc39 February 2020, Oahu Hawaii


XS
TC53
Salesforce
Agoric
MetaMask
Node


XS
TC53
Salesforce
SES
Compartments
Agoric
MetaMask
Node

EcmaScript

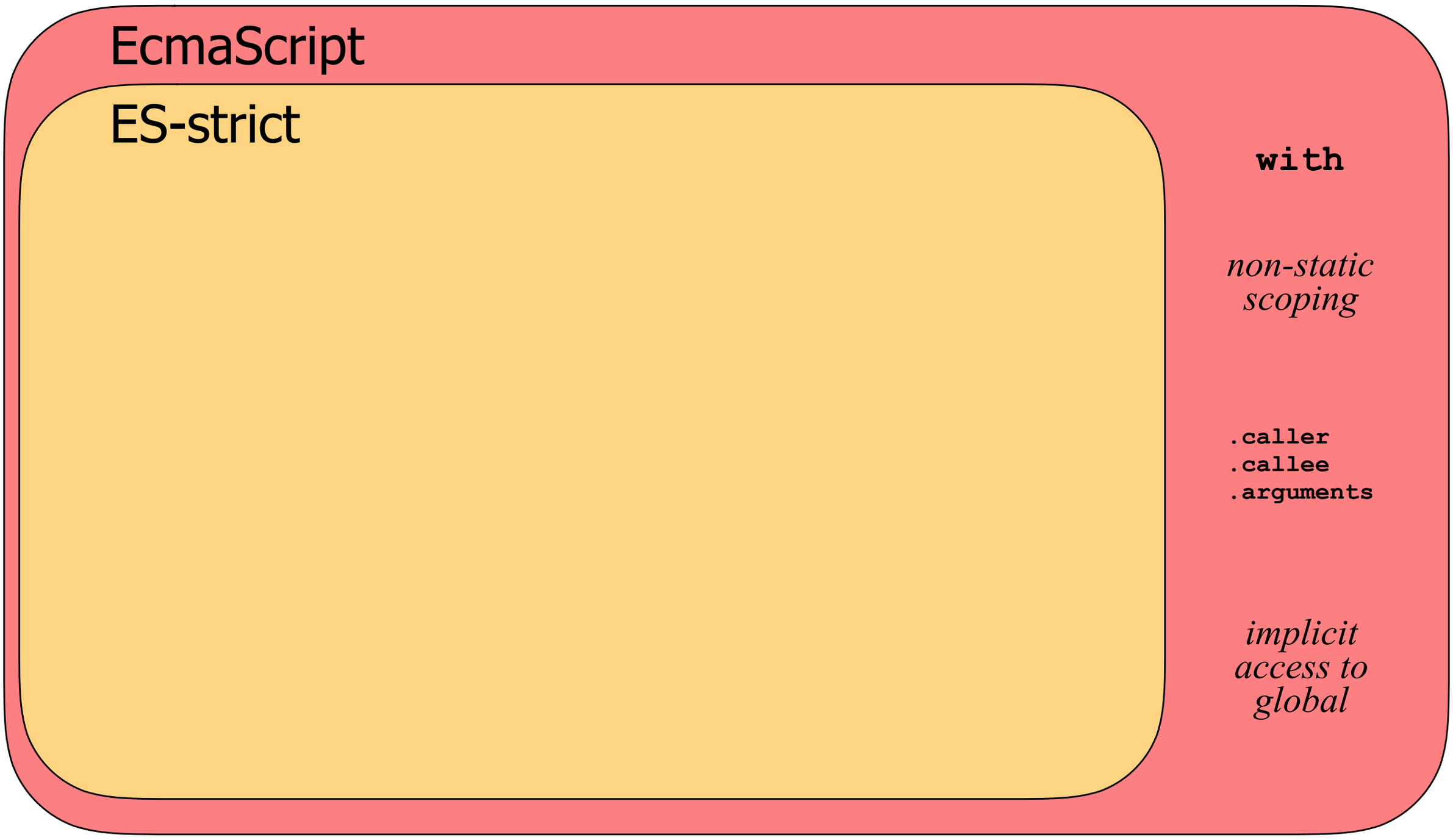
`with`

non-static scoping

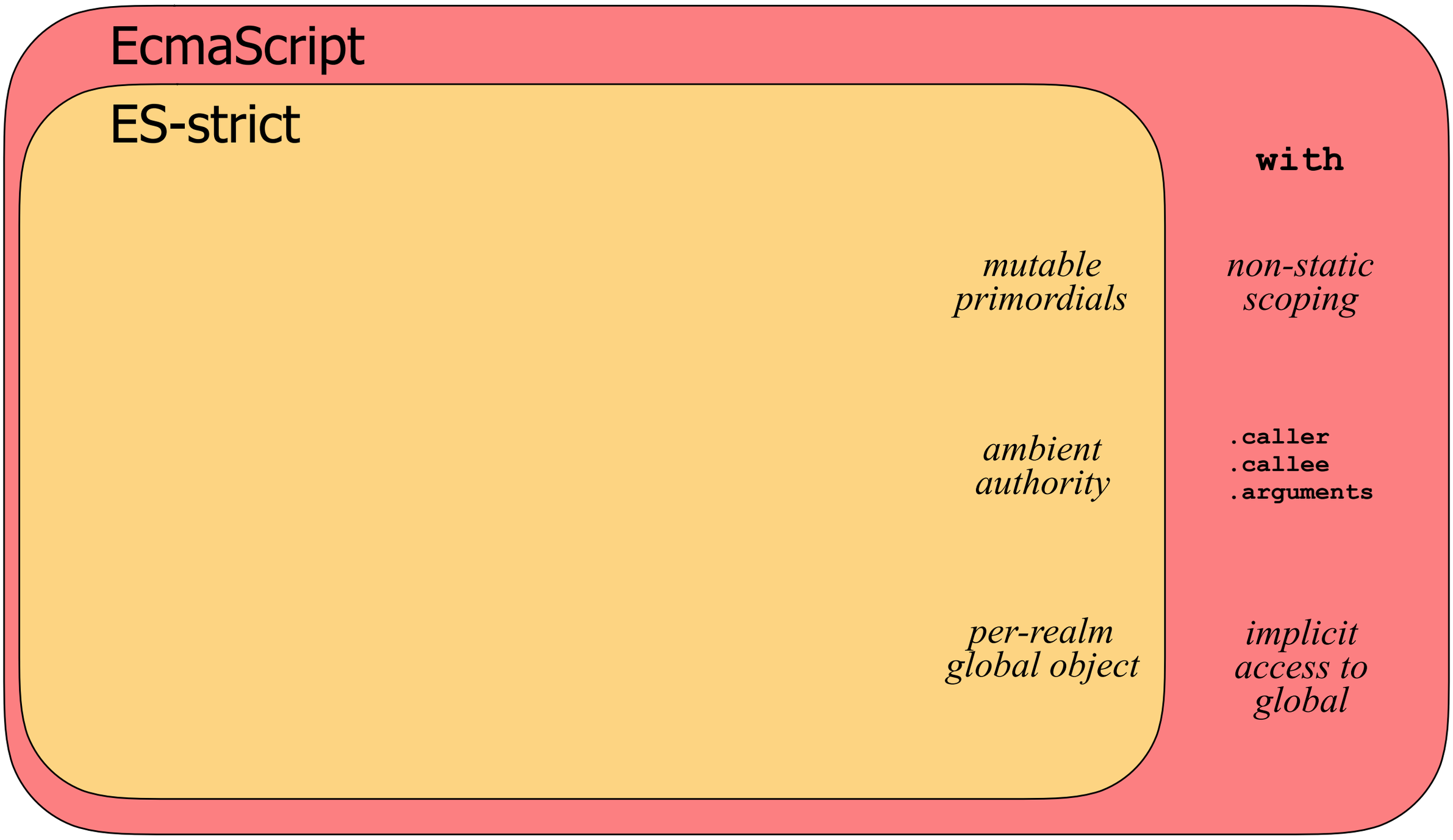
`.caller`
`.callee`
`.arguments`

implicit access to global

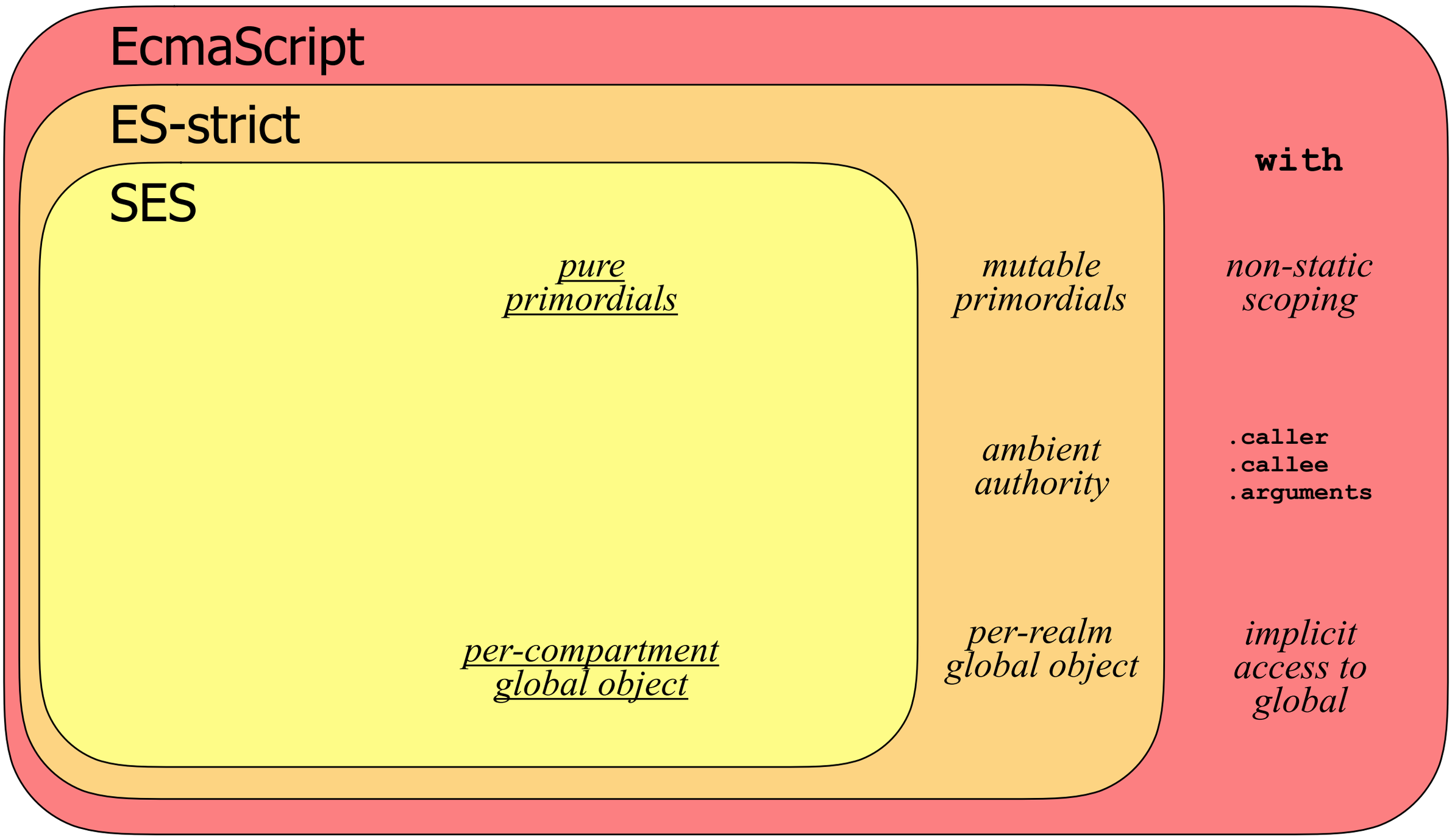
Don’t add security. Remove insecurity.



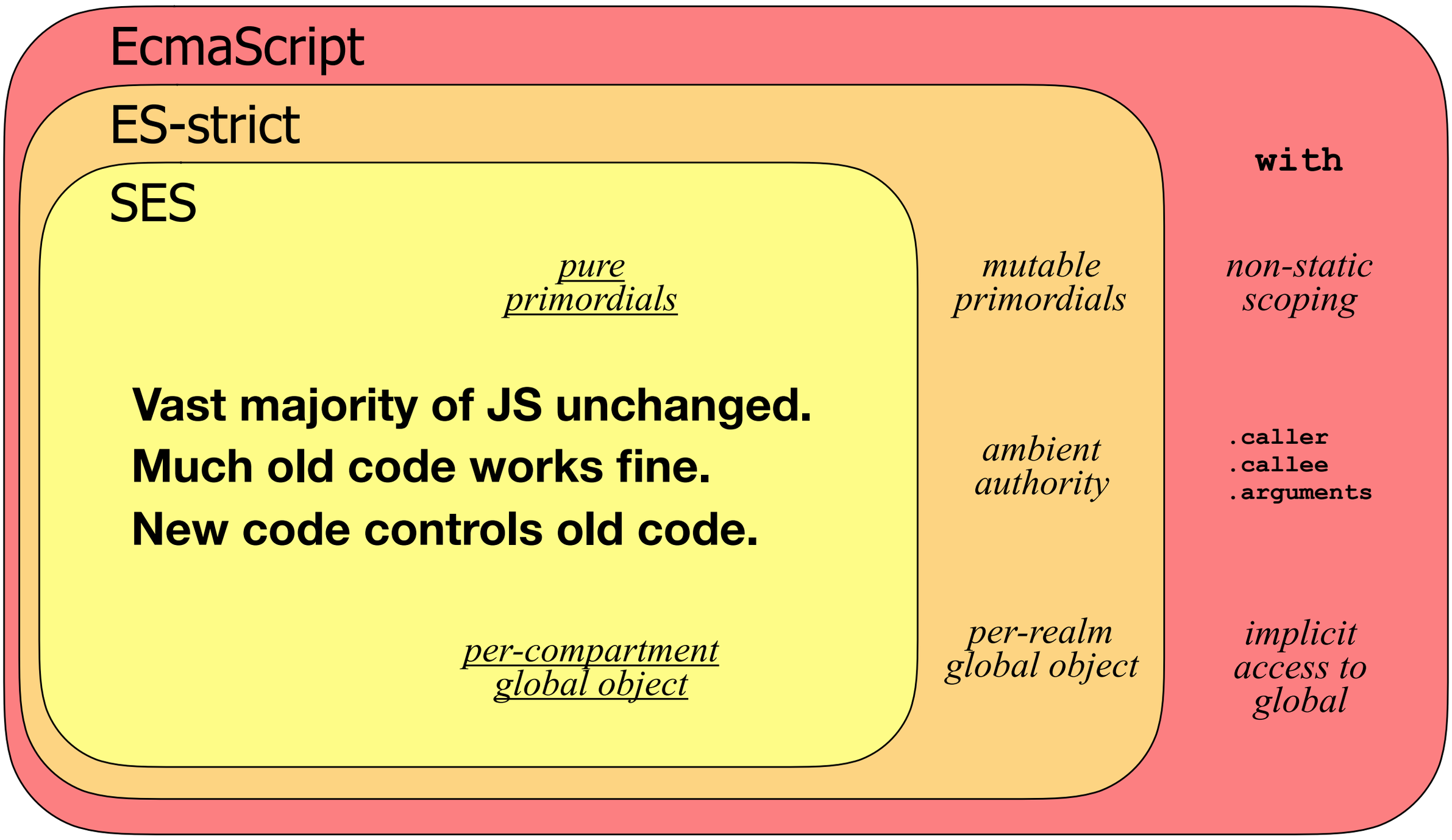
Don’t add security. Remove insecurity.



Don’t add security. Remove insecurity.



Don’t add security. Remove insecurity.



SES

pure primordials

Vast majority of JS unchanged.
Much old code works fine.
New code controls old code.

per-compartment global object

Min memory. Omit needless evaluation

SES

Typical tc53

pure primordials

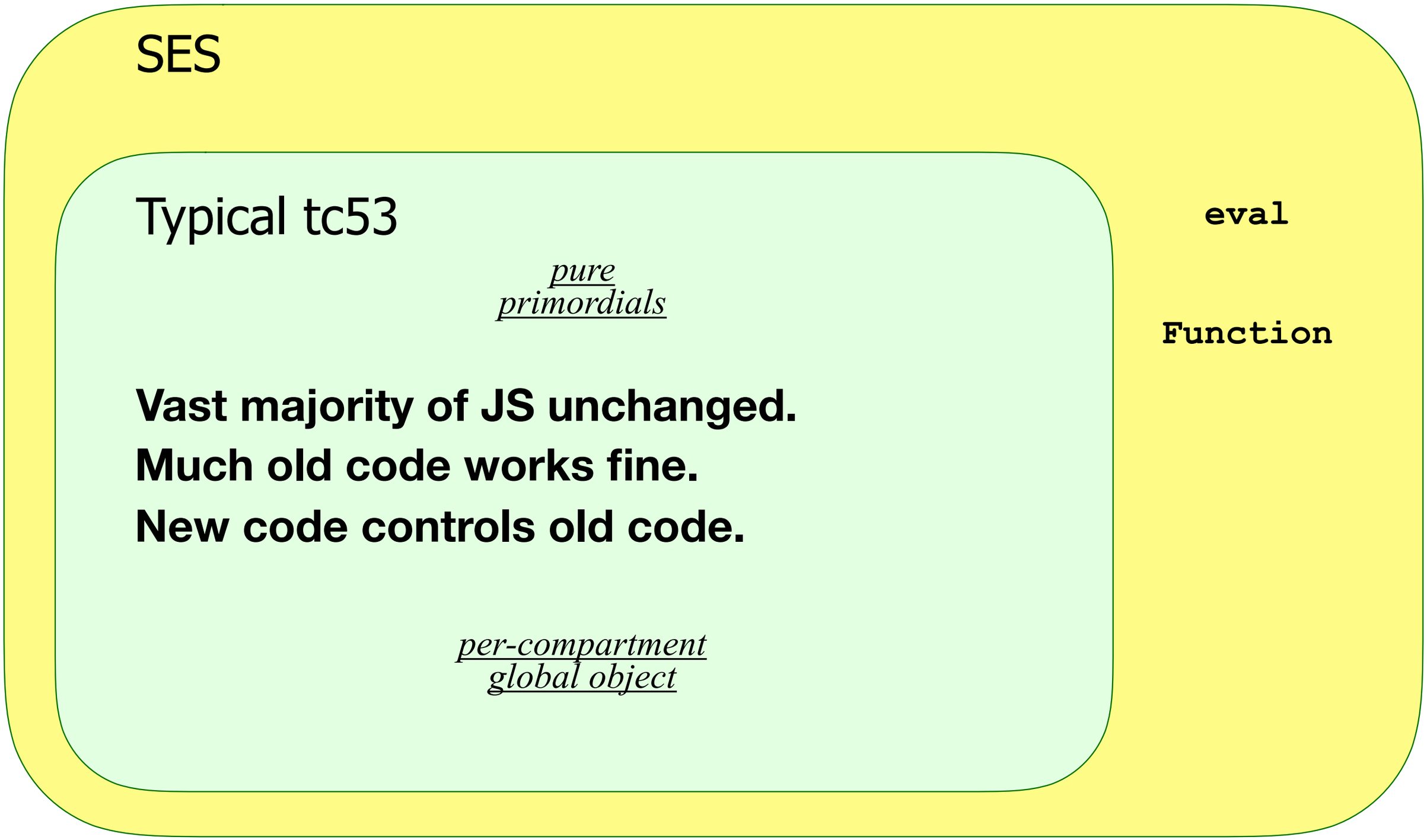
Vast majority of JS unchanged.

Much old code works fine.

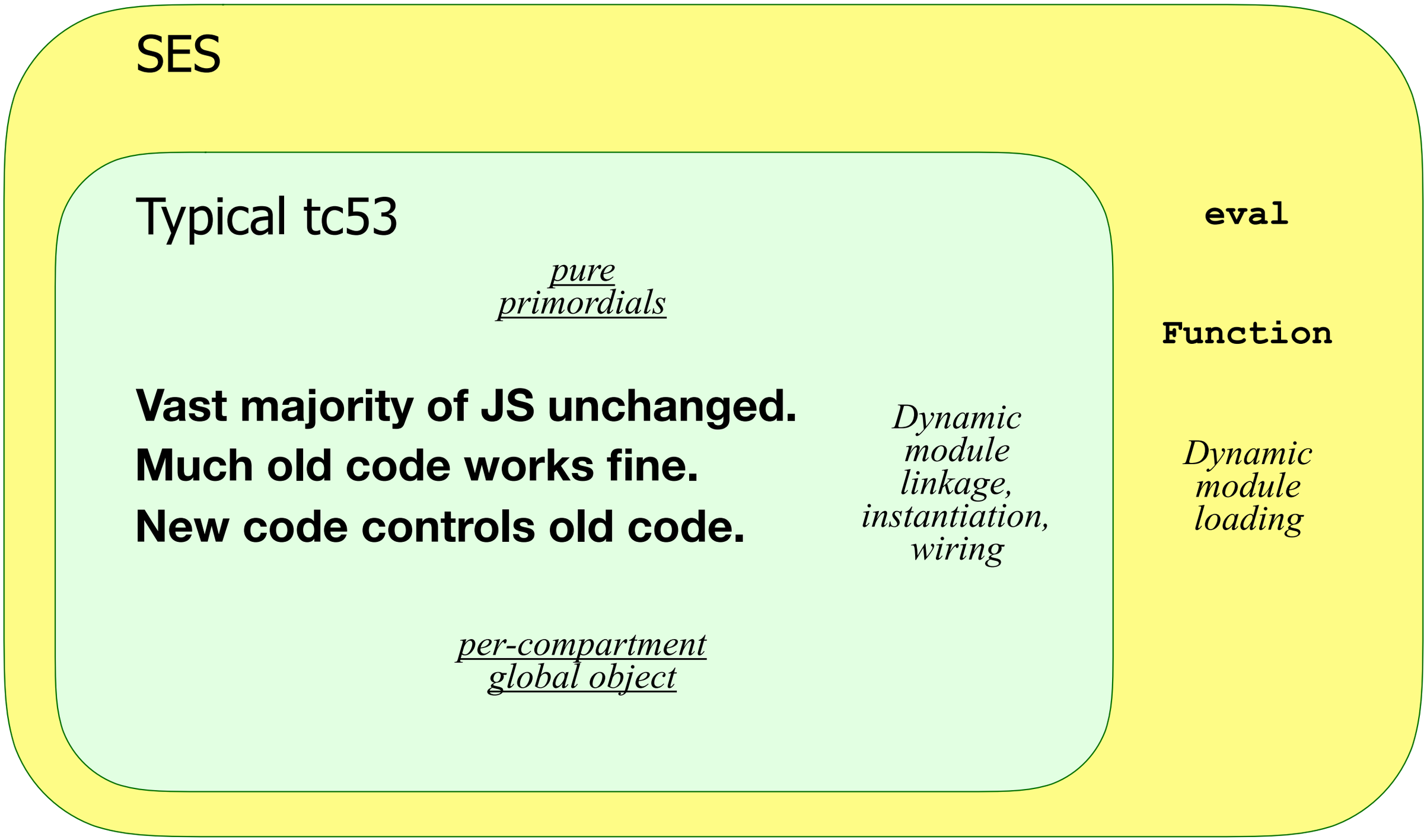
New code controls old code.

per-compartment global object

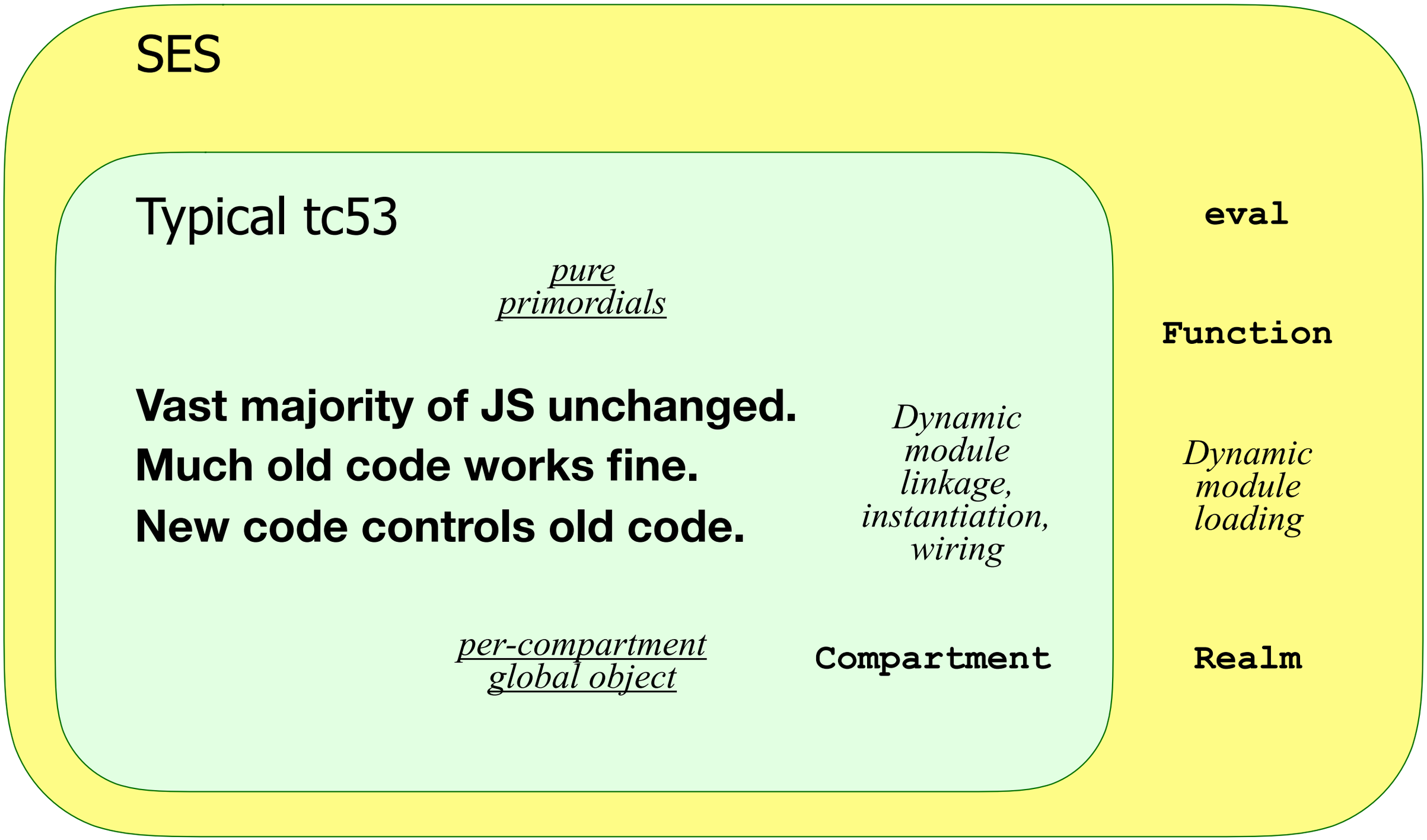
Min memory. Omit needless evaluation



Min memory. Omit needless evaluation



Min memory. Omit needless evaluation



Min memory. Omit needless evaluation

CSP

SES

Typical tc53

pure primordials

Vast majority of JS unchanged.
Much old code works fine.
New code controls old code.

per-compartment global object

Dynamic module linkage, instantiation, wiring

`Compartment`

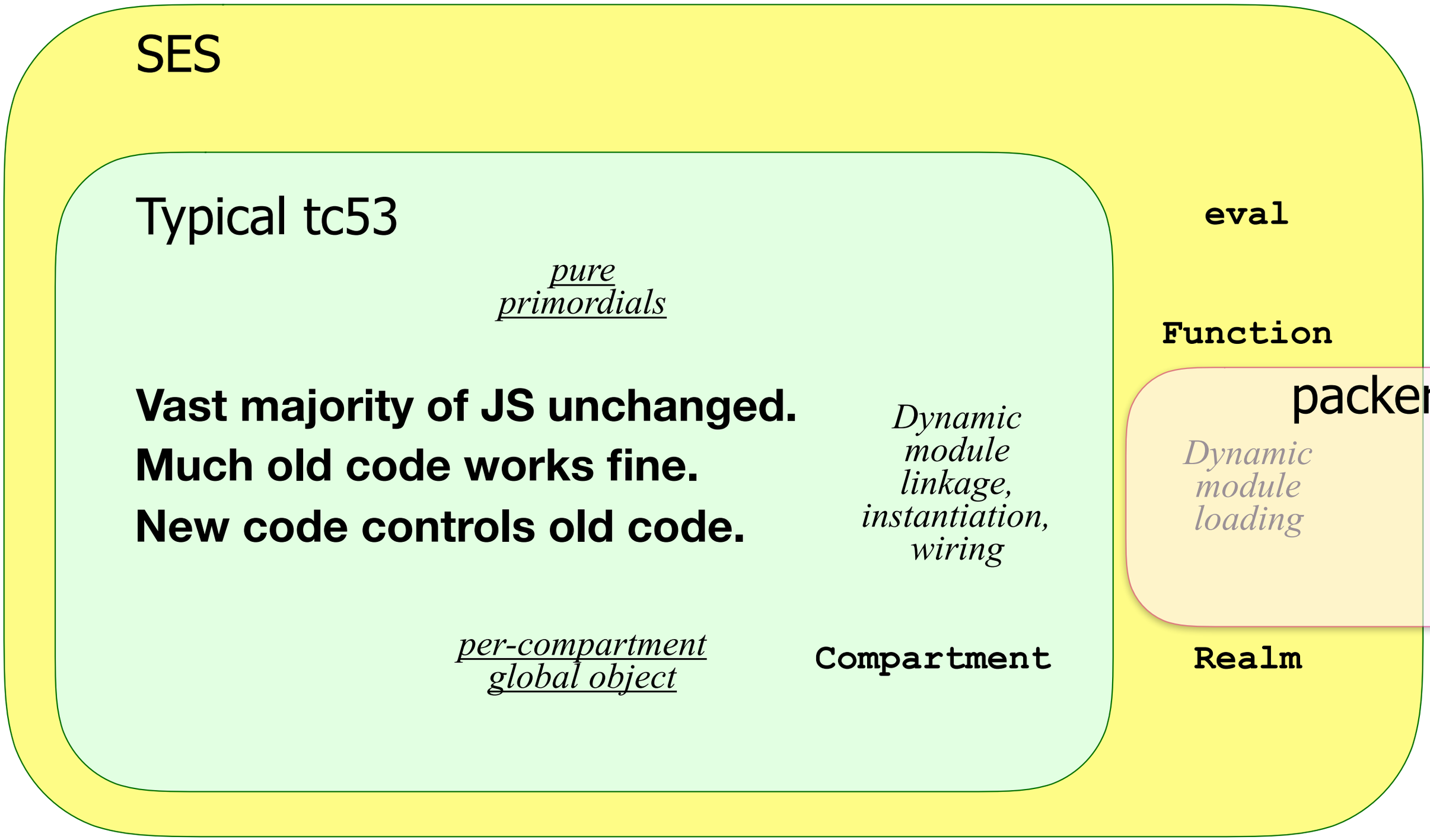
`eval`

`Function`

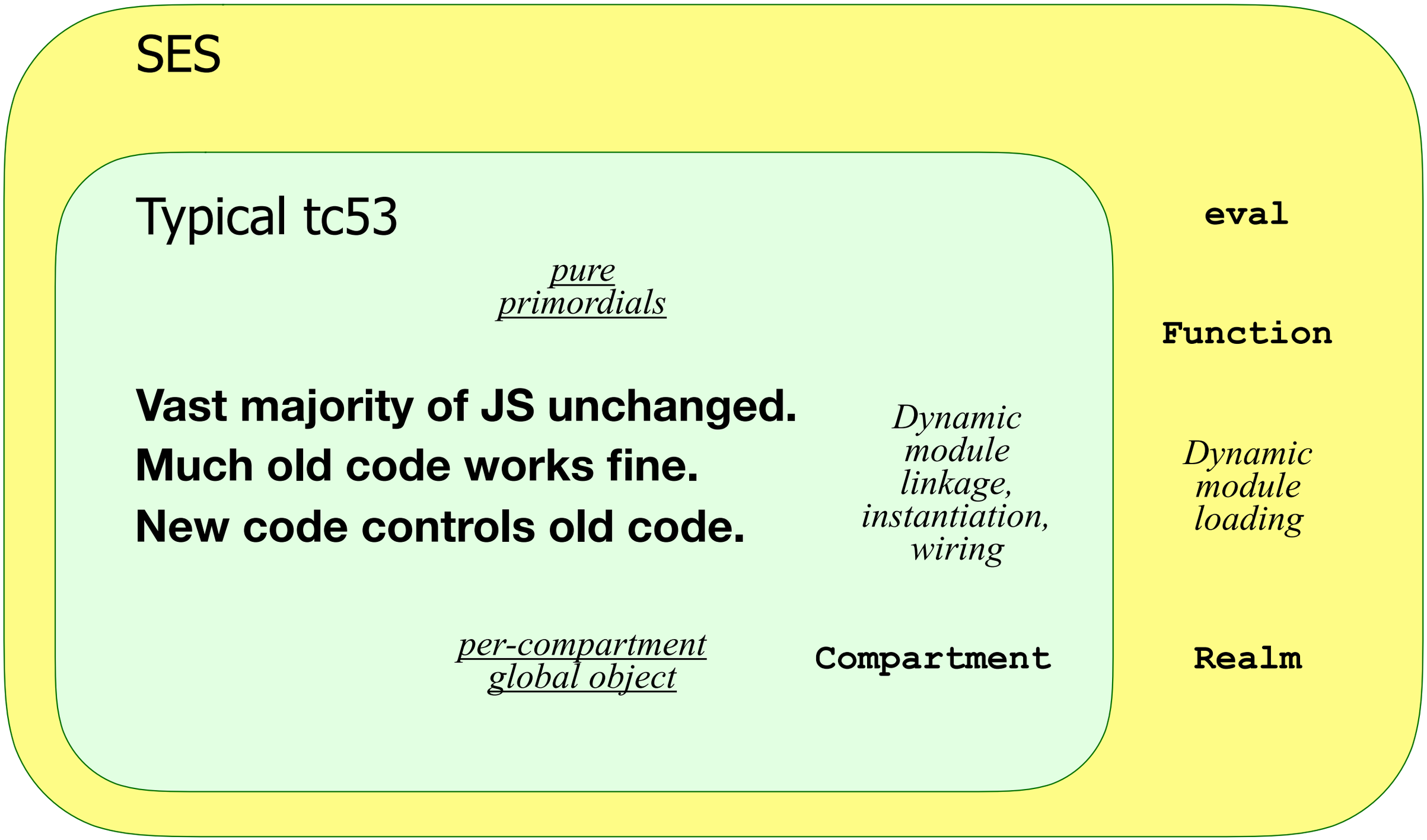
Dynamic module loading

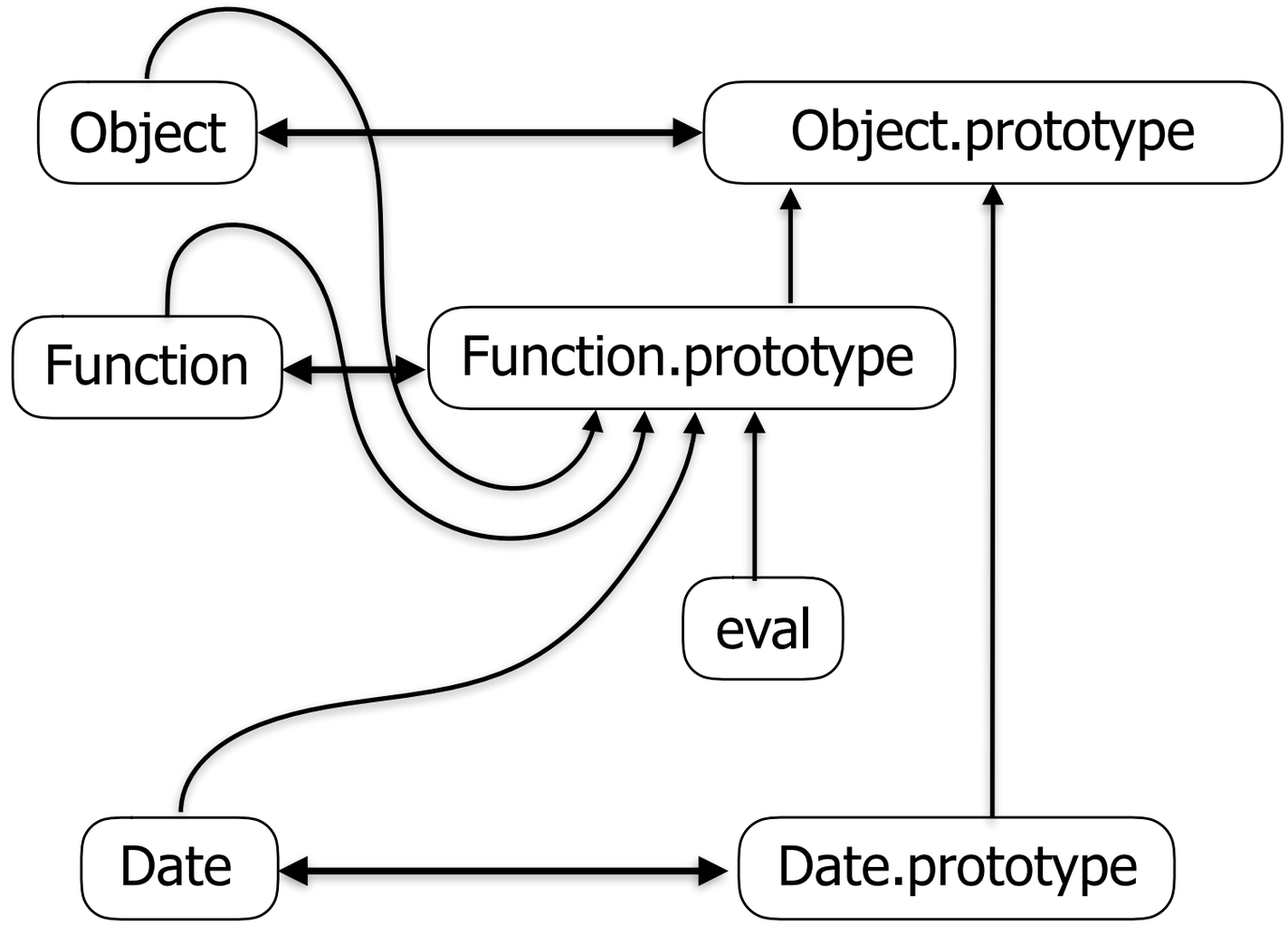
`Realm`

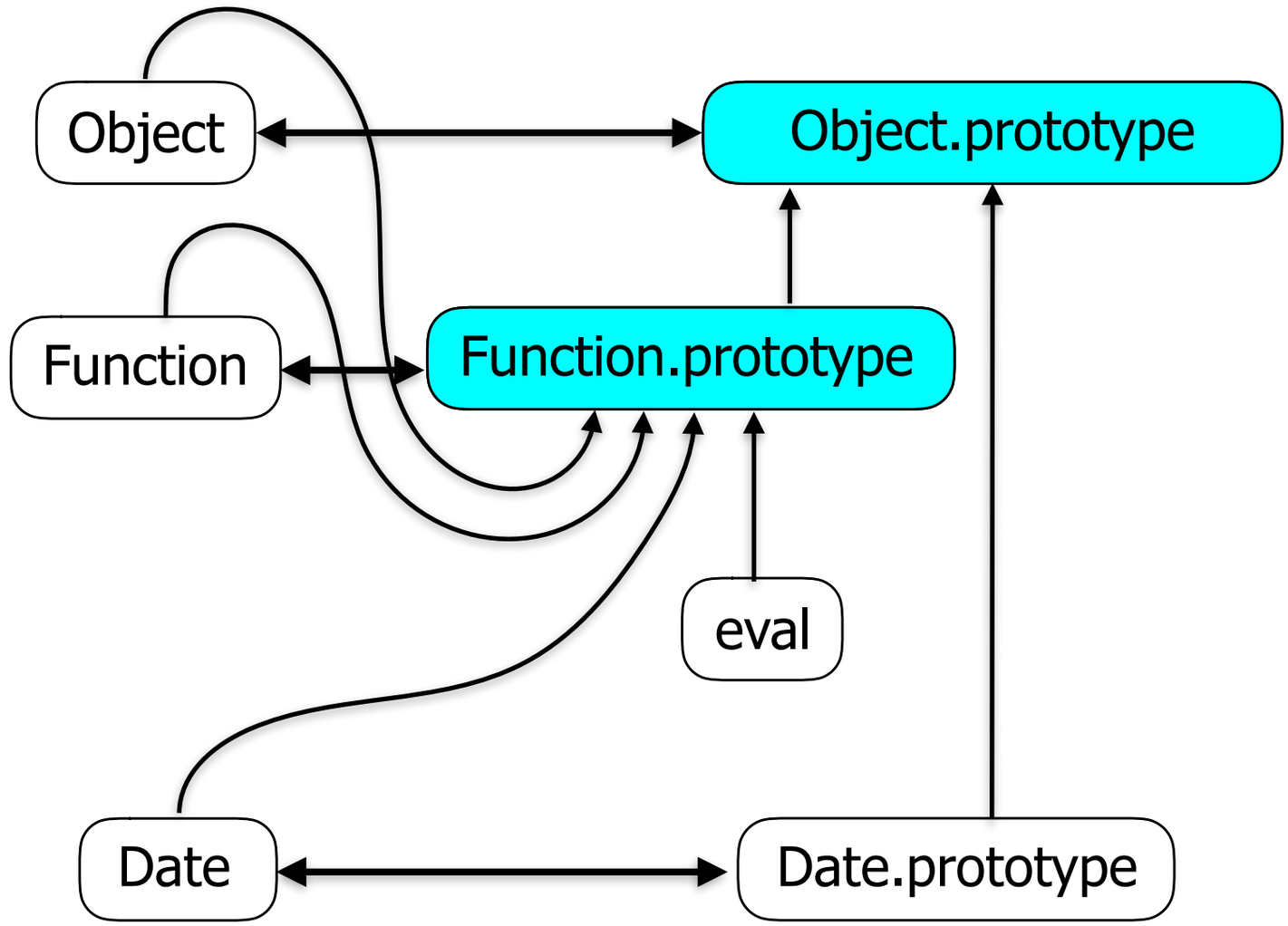
Min memory. Omit needless evaluation

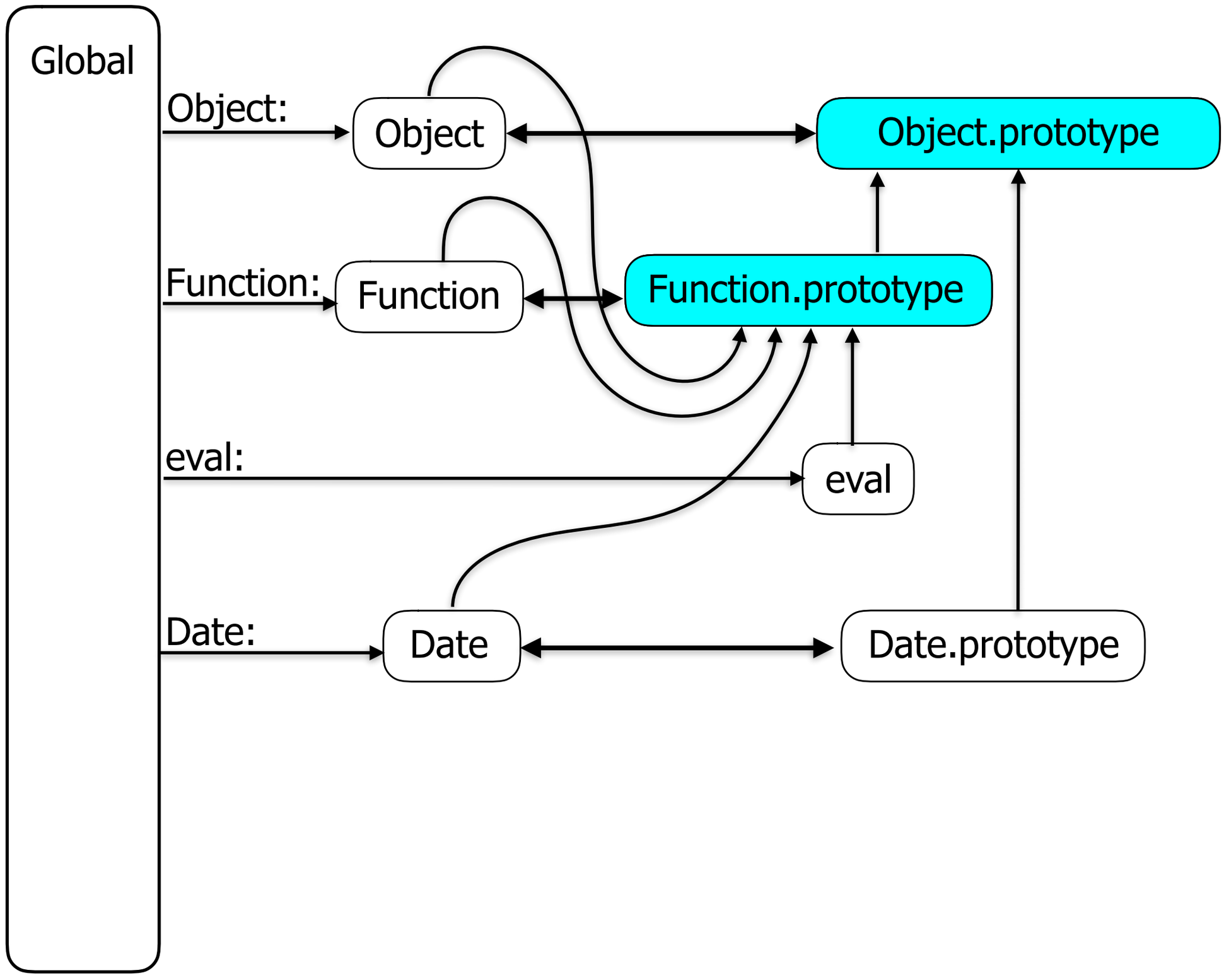


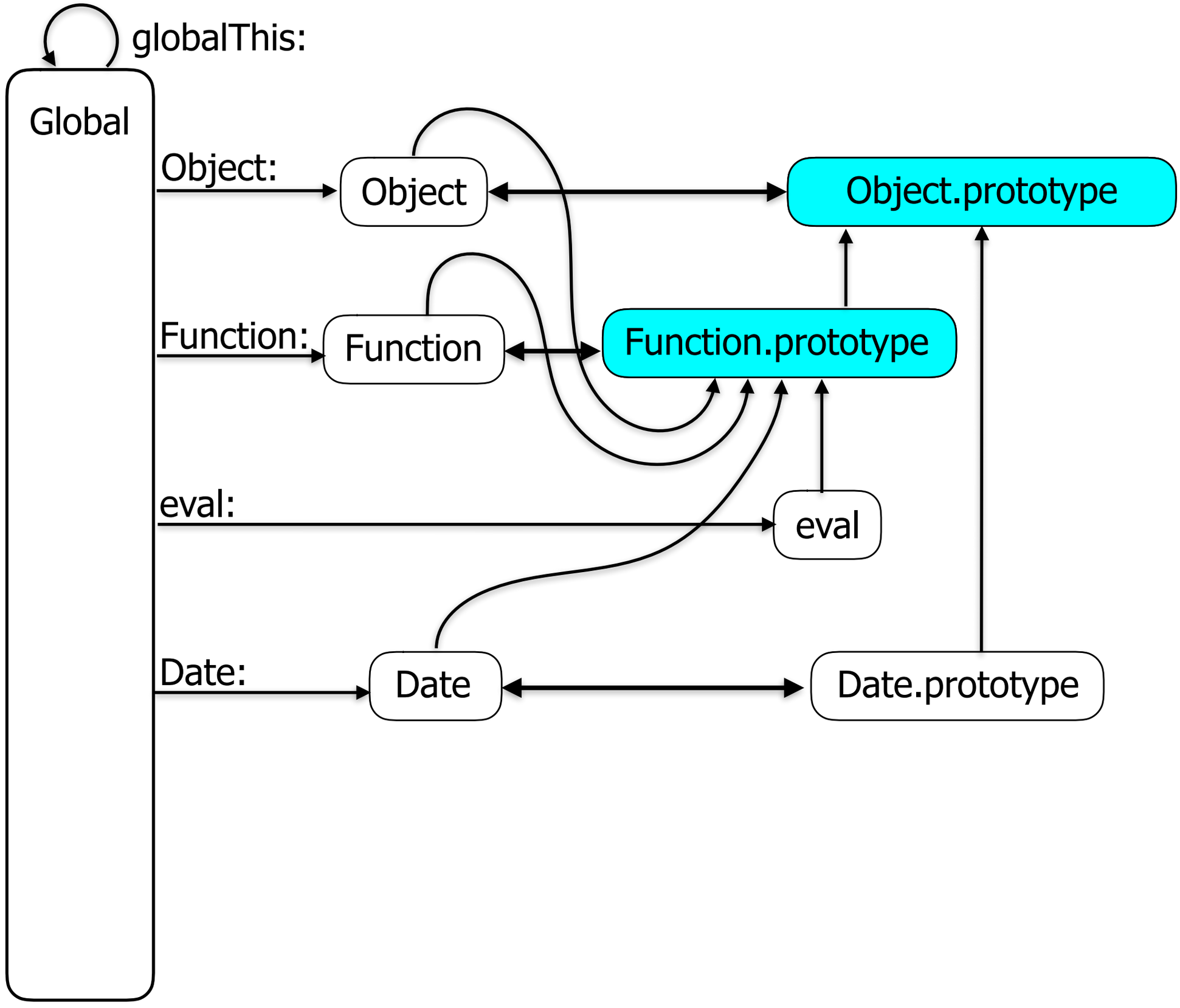
Min memory. Omit needless evaluation

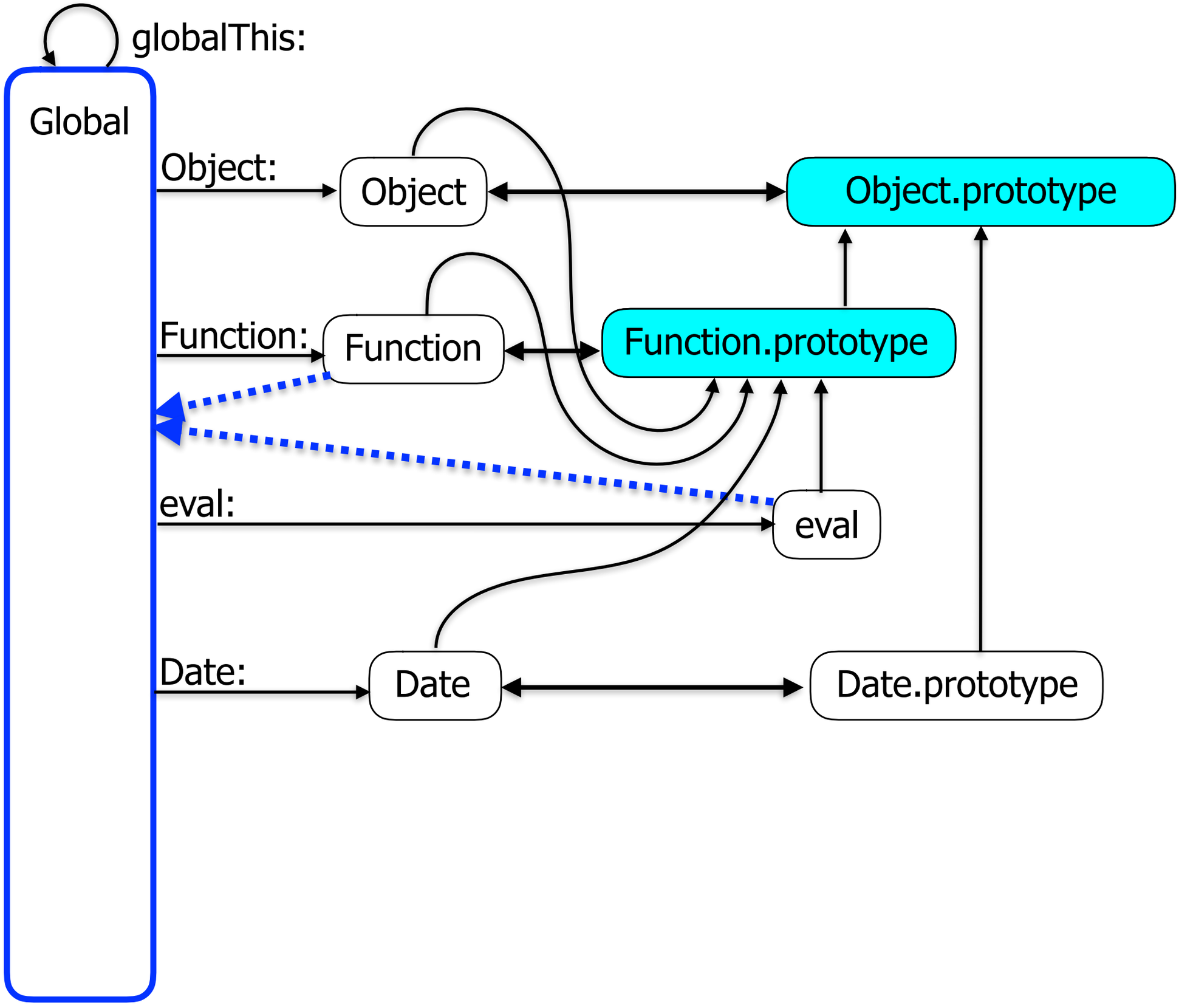


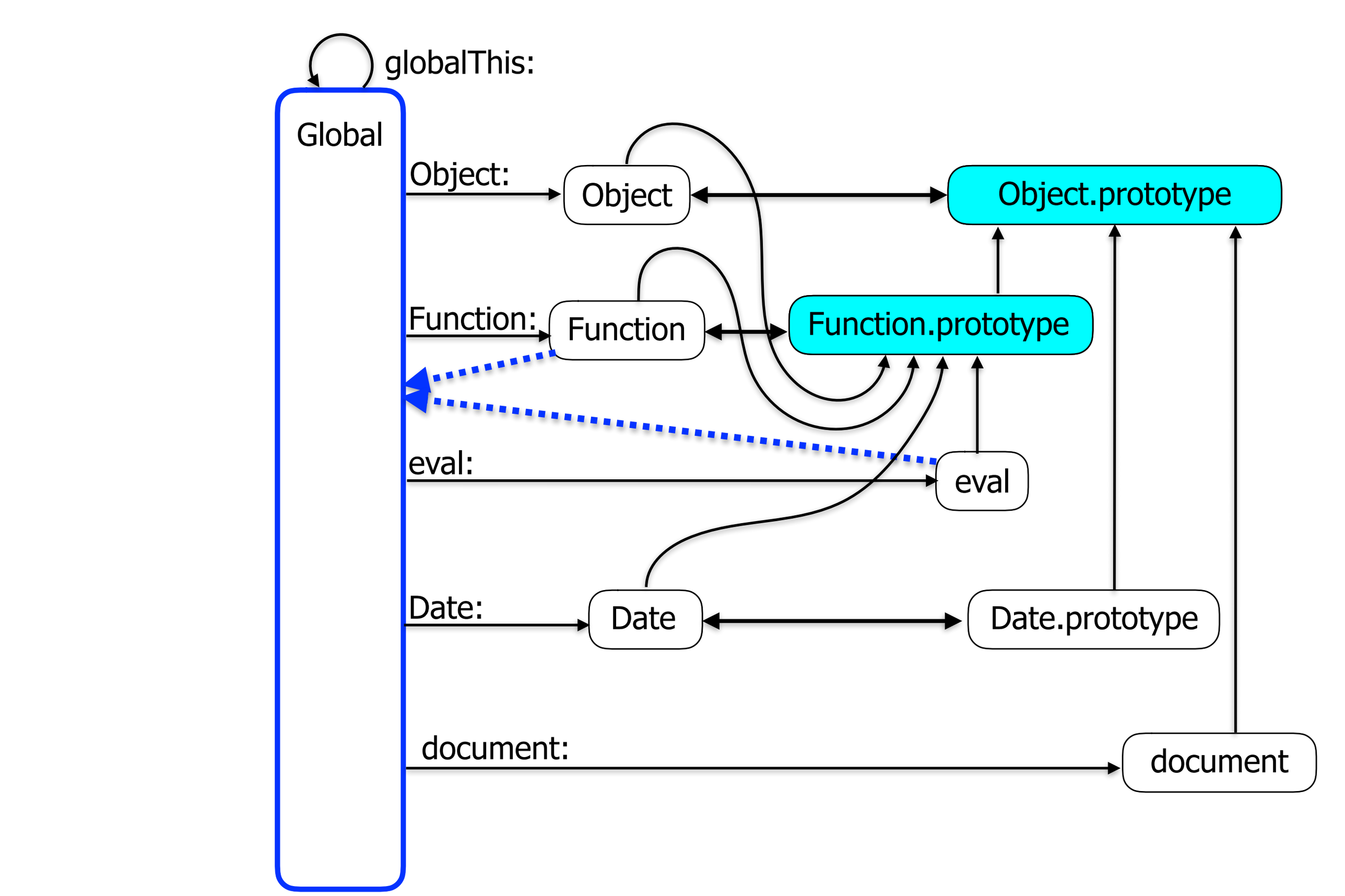

Object
Object.prototype
Function
Function.prototype
eval
Date
Date.prototype

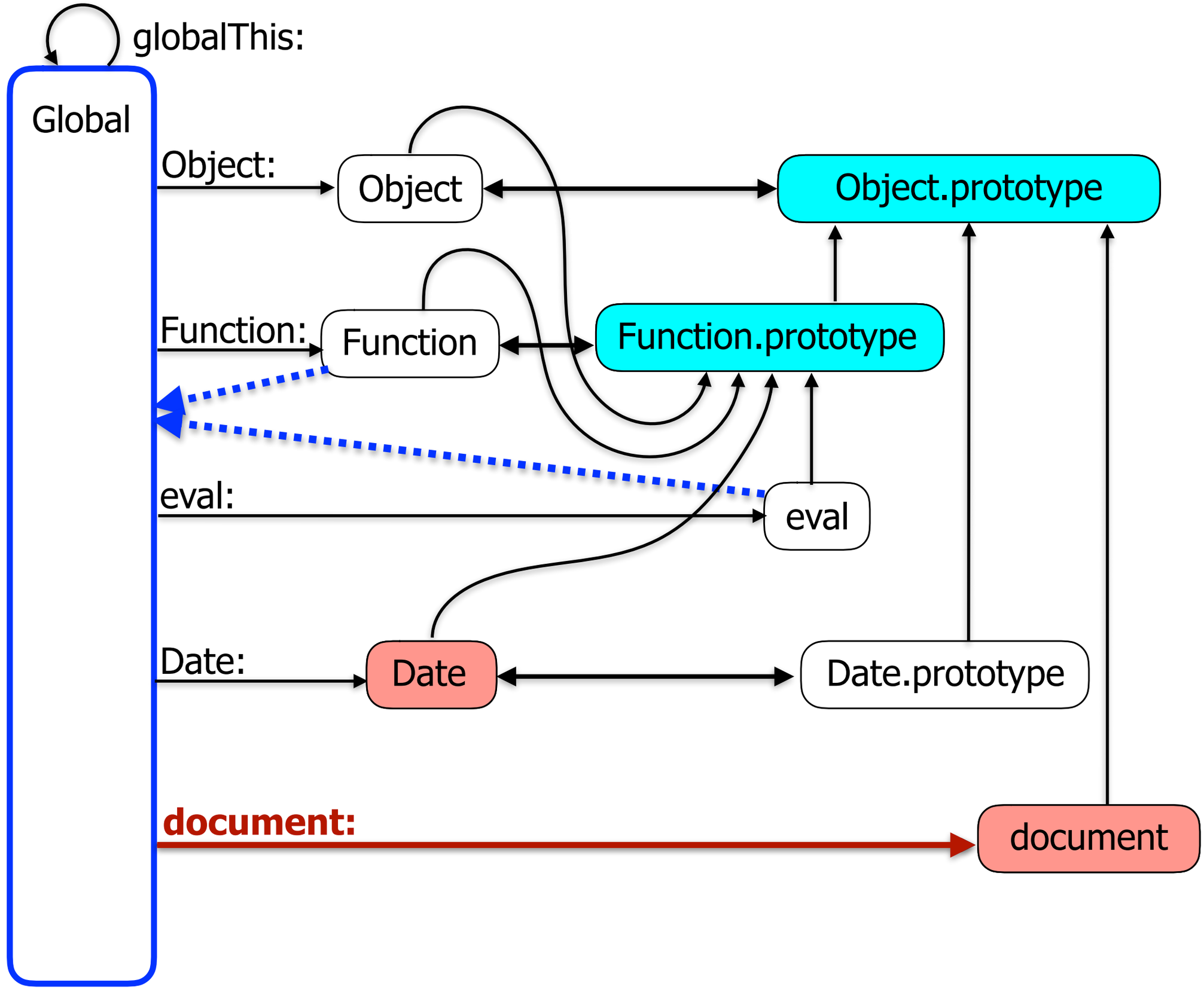

Object
Object.prototype
Function
Function.prototype
eval
Date
Date.prototype

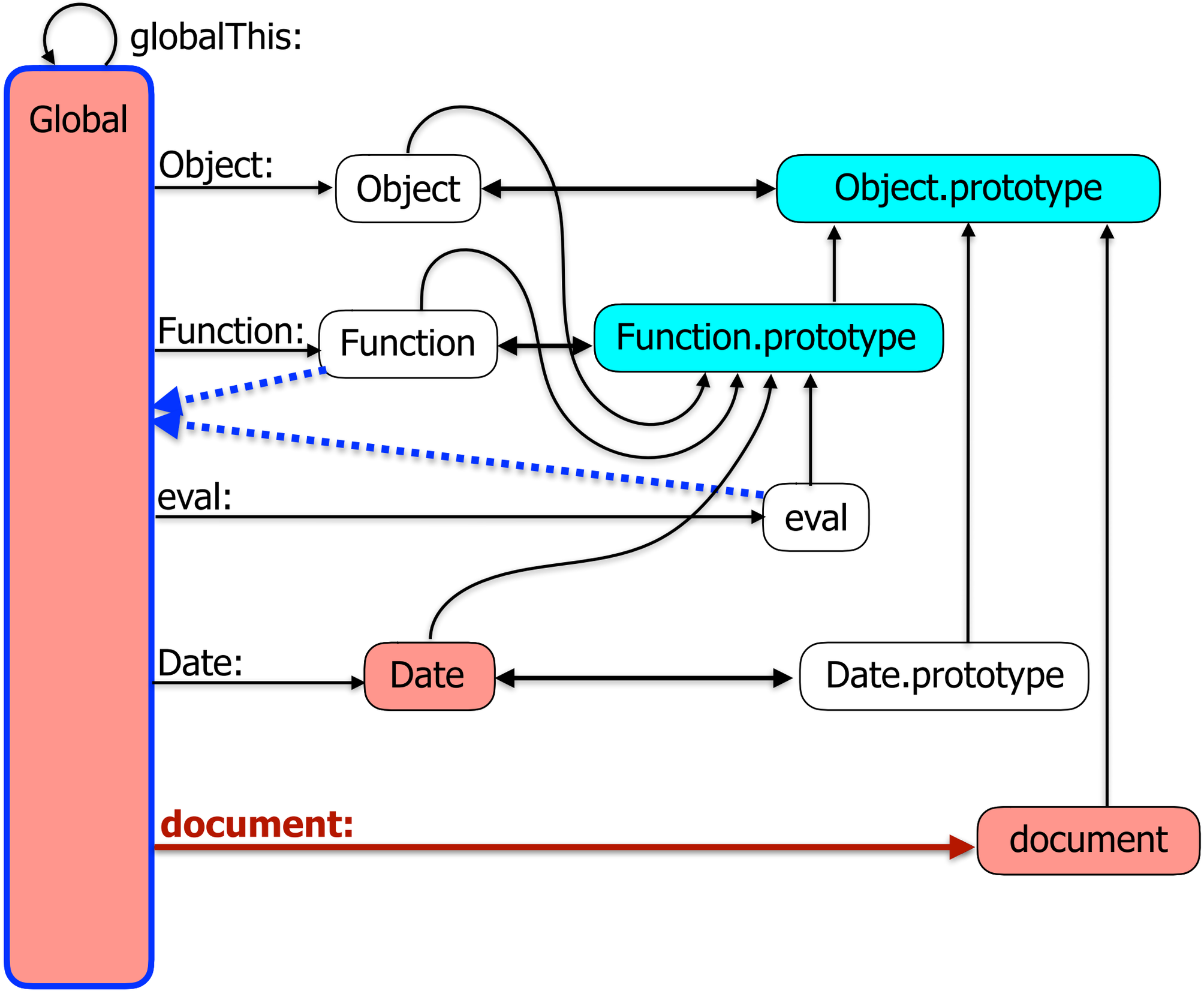

Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype

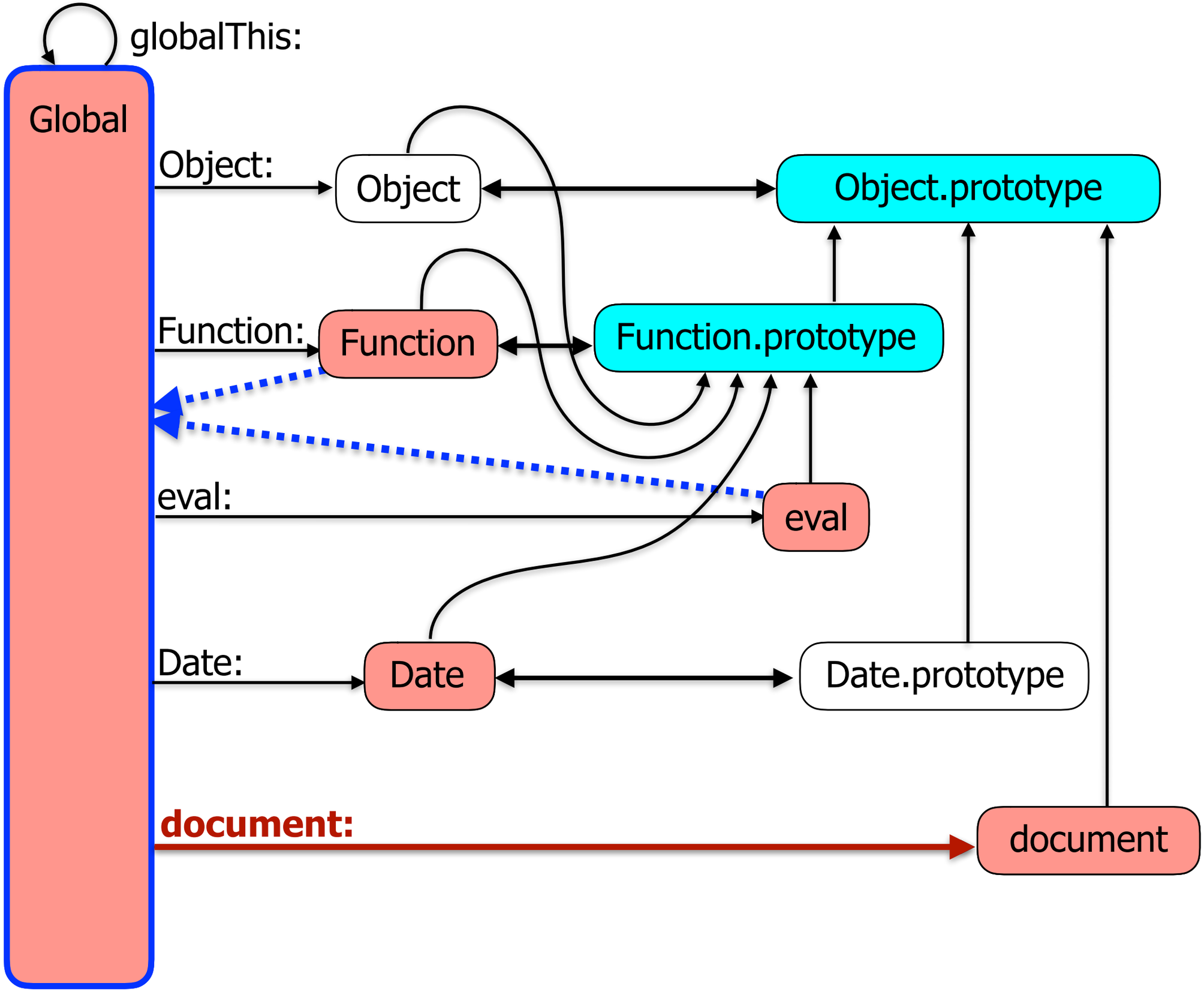

globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype

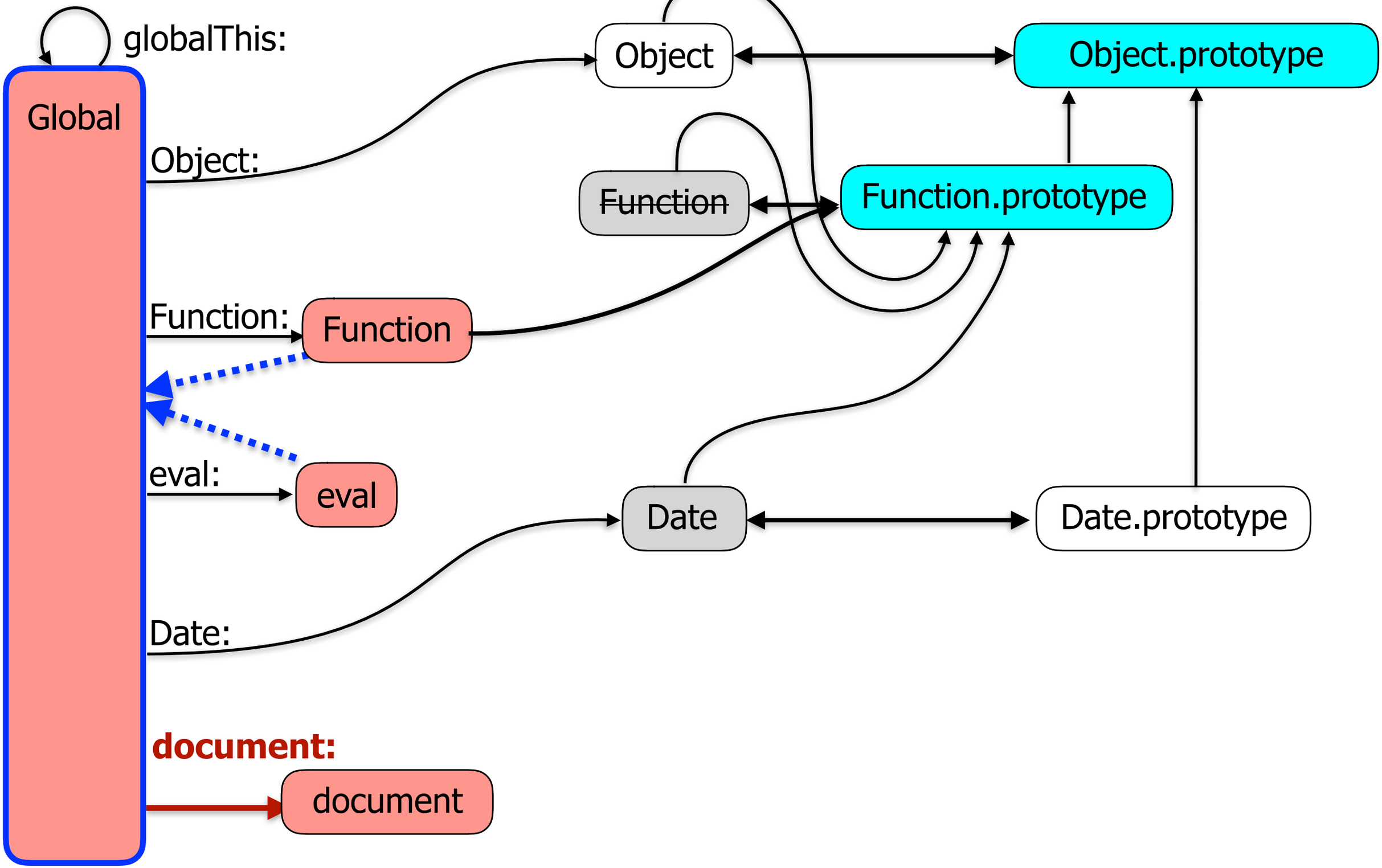

globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype


globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype
document:
document


globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype
document:
document


globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype
document:
document


globalThis:
Global
Object:
Object
Object.prototype
Function:
Function
Function.prototype
eval:
eval
Date:
Date
Date.prototype
document:
document


globalThis:
Global
Object:
Function:
eval:
Date:
document:
Object
Function
Function
eval
Date
document
Object.prototype
Function.prototype
Date.prototype

Start Compartment

globalThis:

Global

Object:

Function:

Function

eval:

eval

Date:

document:

document

Frozen Shared Intrinsics

Object

Object.prototype

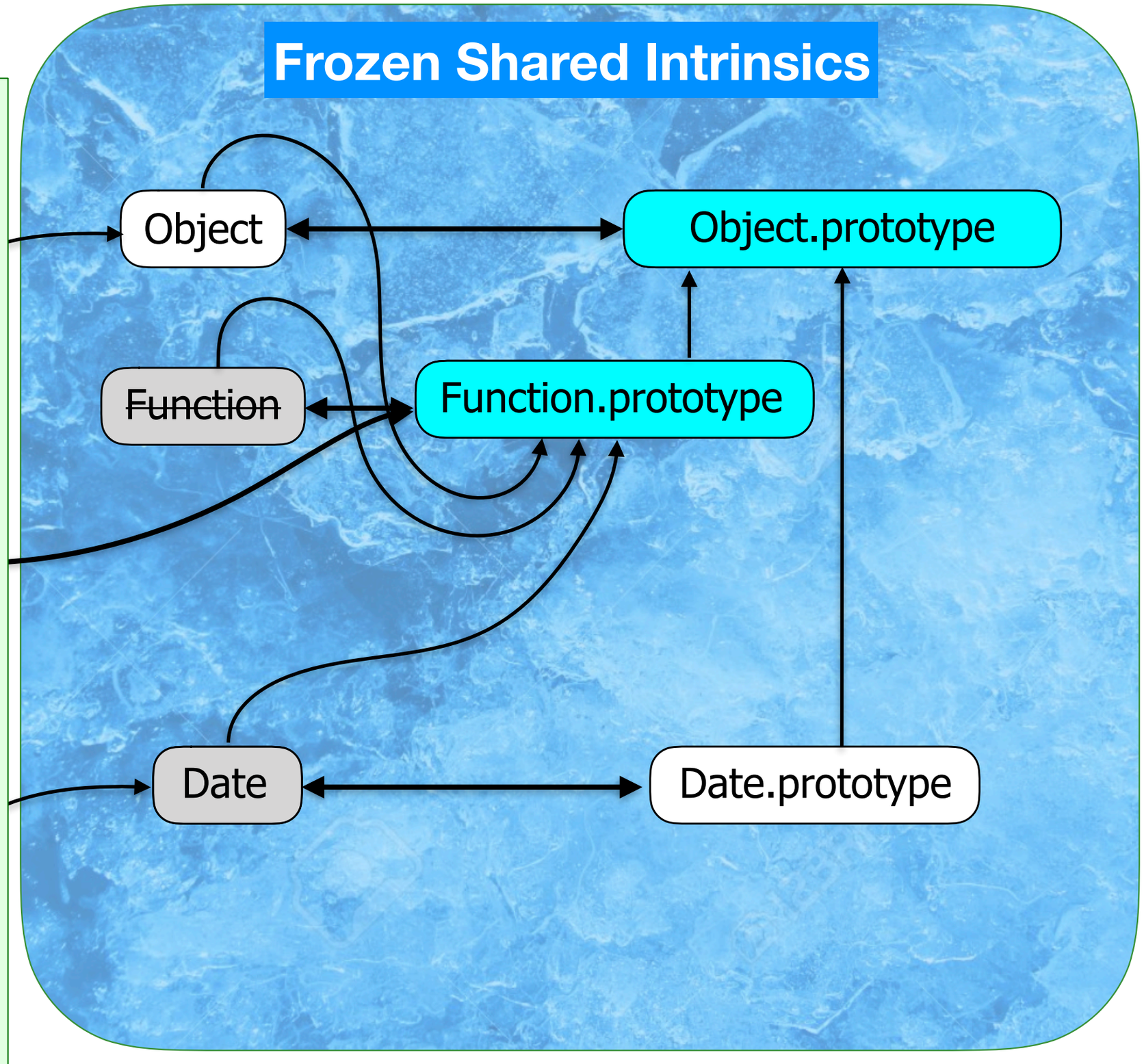
~~Function~~

Function.prototype

Date

Date.prototype

Compartment



Compartment

globalThis:

Global

Object:

Function:

Function

eval:

eval

Date:

Frozen Shared Intrinsics

Object

Object.prototype

~~Function~~

Function.prototype

Date

Date.prototype

Compartment

globalThis:

Global

Object:

Function:

Function

eval:

eval

Date:

process:

process

Frozen Shared Intrinsics

Object

Object.prototype

~~Function~~

Function.prototype

Date

Date.prototype

Compartment

globalThis:

Global

Object:

Function:

Function

eval:

eval

Date:

Compartment:

Compartment

Frozen Shared Intrinsics

Object

Object.prototype

~~Function~~

Function.prototype

Date

Date.prototype

~~Compartment~~

Compartment.prototype

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                  // strict indirect eval
    src: stringable,
    options: object?         // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                    // strict indirect eval
    src: stringable,
    options: object?           // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                  // strict indirect eval
    src: stringable,
    options: object?         // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                    // strict indirect eval
    src: stringable,
    options: object?           // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

ModuleRecord

[[RealmRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

[[Context]]

ModuleRecord

[[RealmRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

[[Context]]

ModuleRecord

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

[[Context]]

ModuleRecord

[[RRecord]] : RRecord?

[[Environment]] : LexicalEnvironment?

[[Namespace]] : ModuleNamespace?

[[HostDefined]] : Any?

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]] : String[]

[[Status]] : unlinked | linking | linked | evaluating | evaluated

[[EvalError]] : AbruptCompletion?

[[DFSIndex]] : Integer?

[[DFSAncestorIndex]] : Integer?

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]] : ParseNode

[[ImportEntries]] : ImportEntry[]

[[*ExportEntries]] : ExportEntry[]

[[Context]] : ExecutionContext

ModuleRecord

[[RRecord]] : RRecord?

[[Environment]] : LexicalEnvironment?

[[Namespace]] : ModuleNamespace?

[[HostDefined]] : Any?

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]] : String[]

[[Status]] : unlinked | linking | linked | evaluating | evaluated

[[EvalError]] : AbruptCompletion?

[[DFSIndex]] : Integer?

[[DFSAncestorIndex]] : Integer?

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]] : ParseNode

[[ImportEntries]] : ImportEntry[]

[[*ExportEntries]] : ExportEntry[]

[[Context]] : ExecutionContext

Mixes concerns.

Messy.

Let’s refactor.

ModuleRecord

[[RRecord]] : RRecord?

[[Environment]] : LexicalEnvironment?

[[Namespace]] : ModuleNamespace?

[[HostDefined]] : Any?

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]] : String[]

[[Status]] : unlinked | linking | linked | evaluating | evaluated

[[EvalError]] : AbruptCompletion?

[[DFSIndex]] : Integer?

[[DFSAncestorIndex]] : Integer?

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]] : ParseNode

[[ImportEntries]] : ImportEntry[]

[[*ExportEntries]] : ExportEntry[]

[[Context]] : ExecutionContext

ModuleRecord

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

[[Context]]

ModuleRecord

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

CyclicModuleRecord

extends ModuleRecord

[[RequestedModules]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

SourceTextModuleRecord

extends CyclicModuleRecord

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

[[Context]]

StaticModuleRecord

[[RequestedModules]]

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ModuleInitialization

[[ModuleInstance]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

[[Context]]

StaticModuleRecord

[[RequestedModules]]

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ModuleInitialization

[[ModuleInstance]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

[[Context]]

StaticModuleRecord

[[RequestedModules]]

[[ECMAScriptCode]]

[[ImportEntries]]

[[*ExportEntries]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ModuleInitialization

[[ModuleInstance]]

[[Status]]

[[EvalError]]

[[DFSIndex]]

[[DFSAncestorIndex]]

[[Context]]

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

RRecord

[[Intrinsics]] : IntrinsicName —> Object

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

RRecord

[[IntrinsicsRecord]] : IntrinsicName —> Object

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

So far, only pure refactoring.

No observable changes.

No extra state.

Cleaner.

Sets the stage for…

ModuleInstance

[[StaticModuleRecord]]

[[RRecord]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[RRecord]]

[[Environment]]

[[HostDefined]]

RRecord

[[Intrinsics]] : IntrinsicName —> Object

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

ModuleInstance

[[StaticModuleRecord]]

[[Compartment]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[Compartment]]

[[Environment]]

[[HostDefined]]

Compartment

[[Intrinsics]] : IntrinsicName —> Object

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

ModuleInstance

[[StaticModuleRecord]]

[[Compartment]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[Compartment]]

[[Environment]]

[[HostDefined]]

Compartment

[[SharedIntrinsics]] : IntrinsicName —> Object // per-Realm

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

ModuleInstance

[[StaticModuleRecord]]

[[Compartment]]

[[Environment]]

[[Namespace]]

[[HostDefined]]

ScriptInstance

[[ECMAScriptCode]]

[[Compartment]]

[[Environment]]

[[HostDefined]]

Compartment

[[SharedIntrinsics]] : IntrinsicName —> Object // per-Realm

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

[[ModuleMap]] : Specifier —> (StaticModuleRecord | ModuleInstance)

[[Hooks]] : Virtual Host Hooks

Compartment

[[SharedIntrinsics]] : IntrinsicName —> Object // per-Realm

[[GlobalObject]] : Object

[[GlobalEnv]] : LexicalEnvironment

[[TemplateMap]] : ParseNode —> Template

[[HostDefined]] : Any?

[[ModuleMap]] : Specifier —> (StaticModuleRecord | ModuleInstance)

[[Hooks]] : Virtual Host Hooks

Compartment

- [[SharedIntrinsics]]
- [[GlobalObject]]
- [[GlobalEnv]]
- [[TemplateMap]]
- [[HostDefined]]
- [[ModuleMap]]
- [[Hooks]]

Compartment

[[SharedIntrinsics]]

[[GlobalObject]]

[[GlobalEnv]]

[[TemplateMap]]

[[HostDefined]]

[[ModuleMap]]

[[Hooks]]

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                // strict indirect eval
    src: stringable,
    options: object?       // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

Compartment

[[SharedIntrinsics]]

[[GlobalObject]]

[[GlobalEnv]]

[[TemplateMap]]

[[HostDefined]]

[[ModuleMap]]

[[Hooks]]

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                // strict indirect eval
    src: stringable,
    options: object?       // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

Compartment

[[SharedIntrinsics]]

[[GlobalObject]]

[[GlobalEnv]]

[[TemplateMap]]

[[HostDefined]]

[[ModuleMap]]

[[Hooks]]

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                // strict indirect eval
    src: stringable,
    options: object?       // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

Compartment

[[SharedIntrinsics]]

[[GlobalObject]]

[[GlobalEnv]]

[[TemplateMap]]

[[HostDefined]]

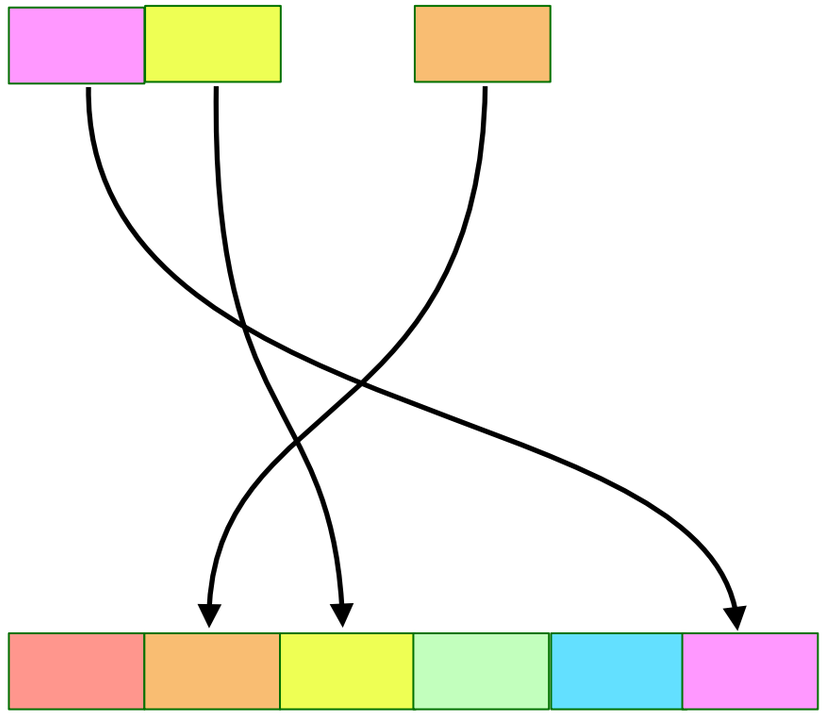
[[ModuleMap]]

[[Hooks]]

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                  // strict indirect eval
    src: stringable,
    options: object?         // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

virtual address

MMU



physical address

physical page

virtual page

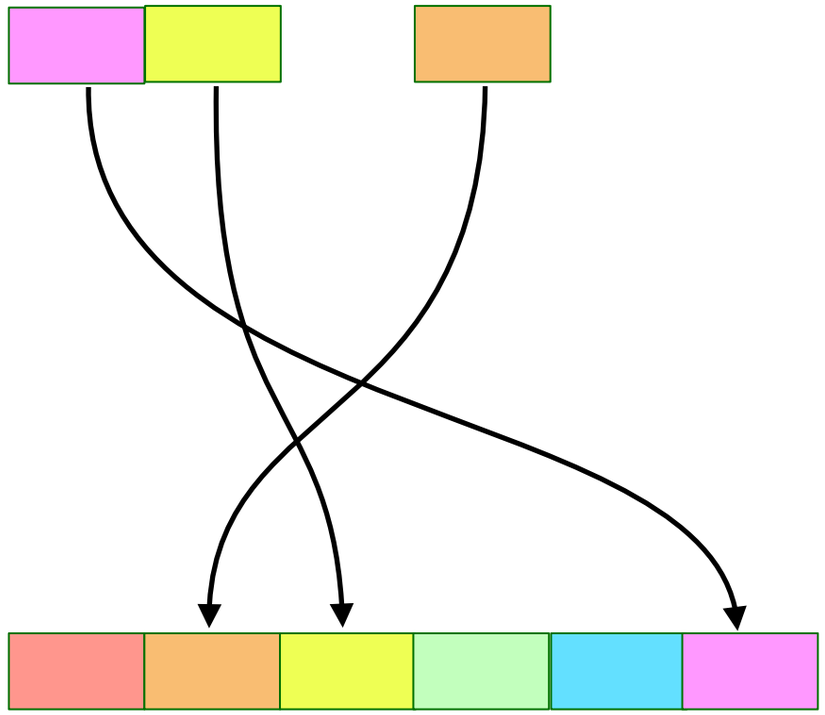
trap

trap trap

child specifier

Module Map

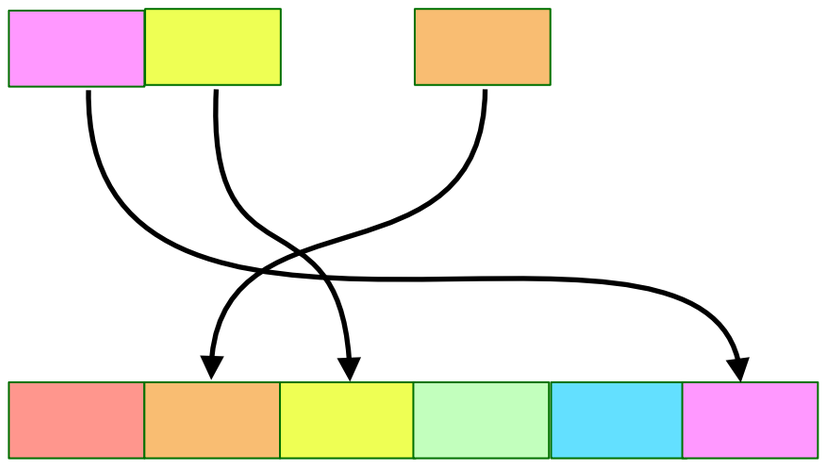
parent specifier



loaded static module

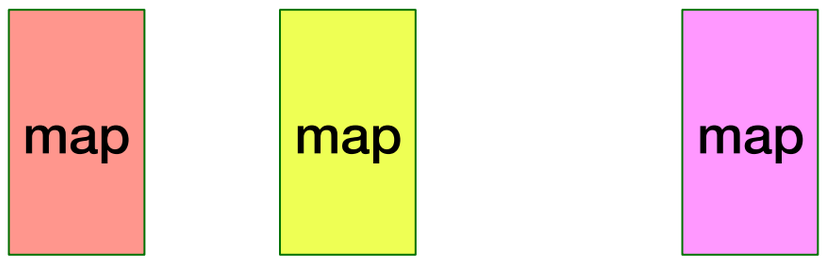
Module loader load load load

child specifier



parent specifier

loaded static module

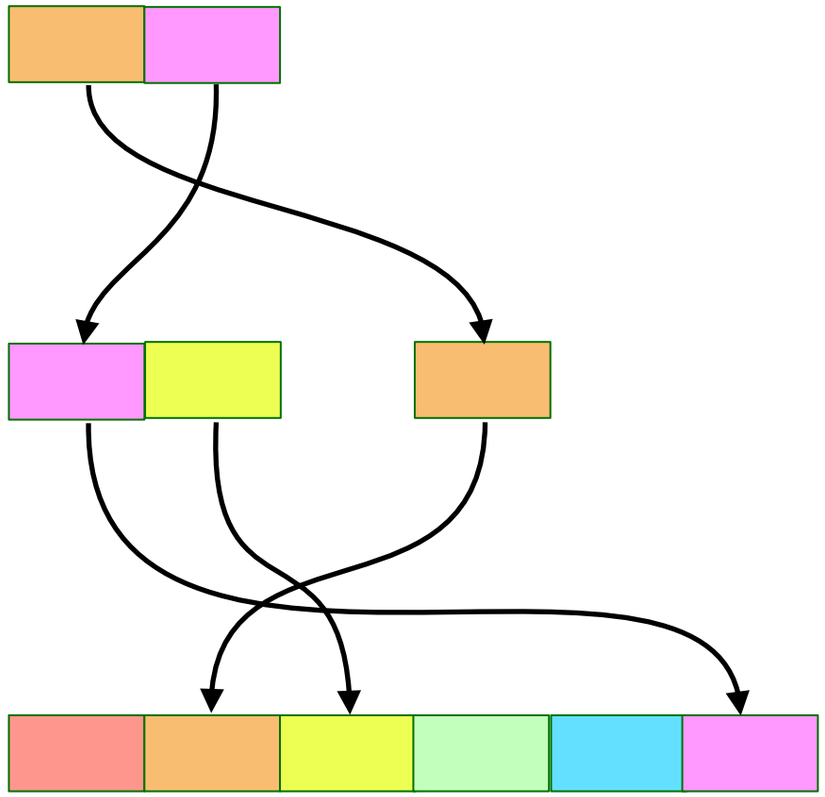


Module loader load load load

grandchild specifier

child specifier

parent specifier



loaded static module

map map map

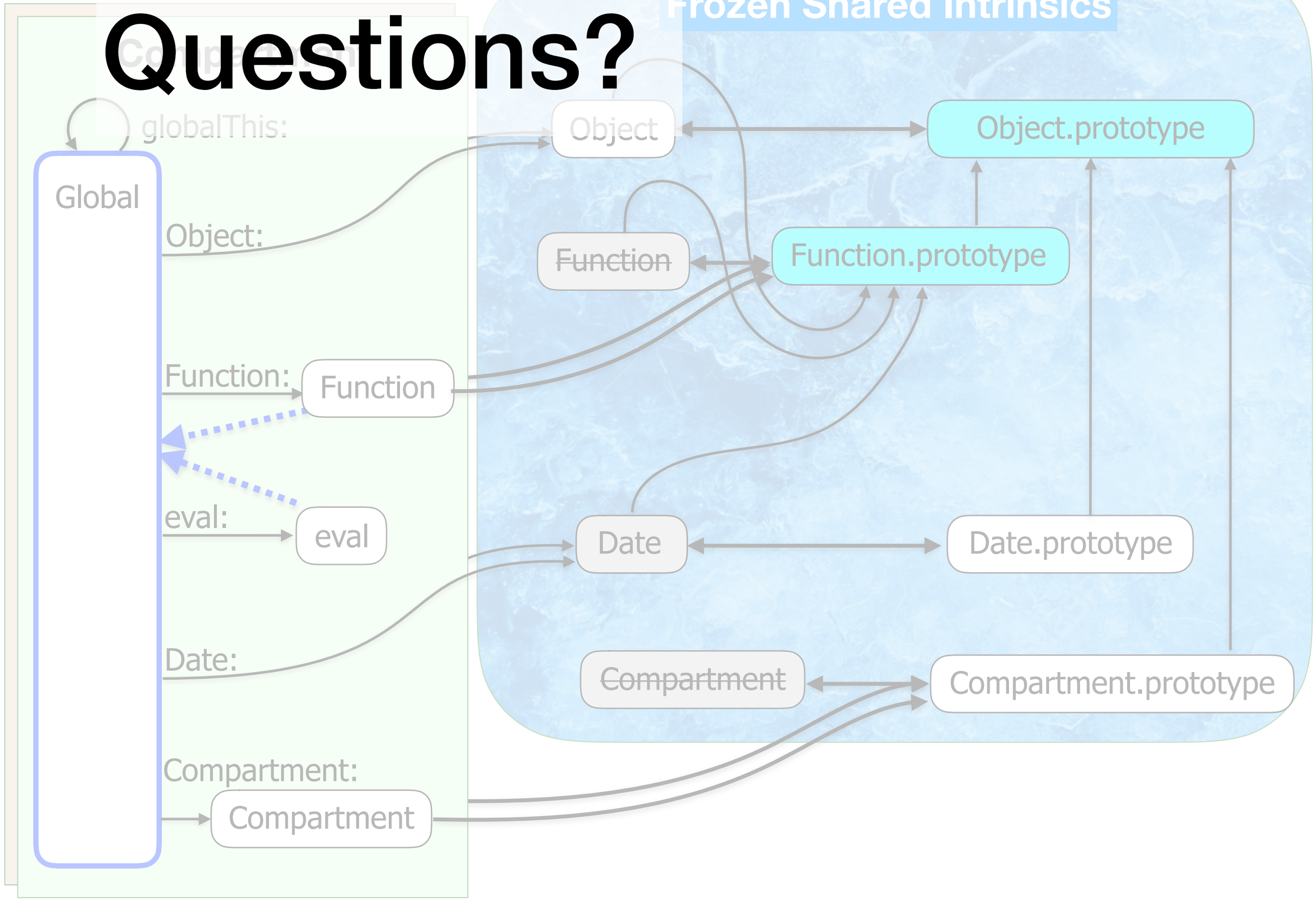
Module loader load load load

Compartment

- [[SharedIntrinsics]]
- [[GlobalObject]]
- [[GlobalEnv]]
- [[TemplateMap]]
- [[HostDefined]]
- **[[ModuleMap]] : Specifier —> (StaticModuleRecord | ModuleInstance)**
- [[Hooks]]

```
class Compartment {
  constructor: (
    endowments: object?, // extra globals
    moduleMap: object?,  // name -> (name | modNS)
    options: object?     // including hooks
  ) -> object

  get global -> object

  evaluate(                // strict indirect eval
    src: stringable,
    options: object?     // per-evaluation
  ) -> any

  importSync(specifier: string) -> ModuleNamespace
  async import(specifier: string) -> promise
}
```

Questions?


Compartment
globalThis:
Global
Object:
Function:
Function
eval:
eval
Date:
Compartment:
Compartment
Frozen Shared Intrinsics
Object
Object.prototype
Function
Function.prototype
Date
Date.prototype
Compartment
Compartment.prototype

Positive feedback from framework authors #36

Open **littledan** opened this issue 19 hours ago · 0 comments

littledan commented 19 hours ago Member + Tip ...

We discussed Realms and SES in the framework outreach group. See minutes at https://github.com/js-outreach/js-outreach-groups/blob/master/frameworks/notes-2020-01-16.md#realms-and-ses . Framework authors were interested in lightweight isolation mechanisms.

Realms and SES under discussion. Are these relevant for frameworks?

- …
- **All: would love to have this**
- …: Realms are good with multiple big teams …
- …
- …: We're using a Worker [in AMP], would love to have a Realm
- … globals isolation would be great.
- … multiple contexts … same frozen realm without interacting, ….
- … great to be able to construct something frozen and secure, **where you can depend on its behavior**.

- … compat issues, e.g., monkey-patching of built-ins would break?
- … **useful … to expose dependencies that tamper with globals**
- …