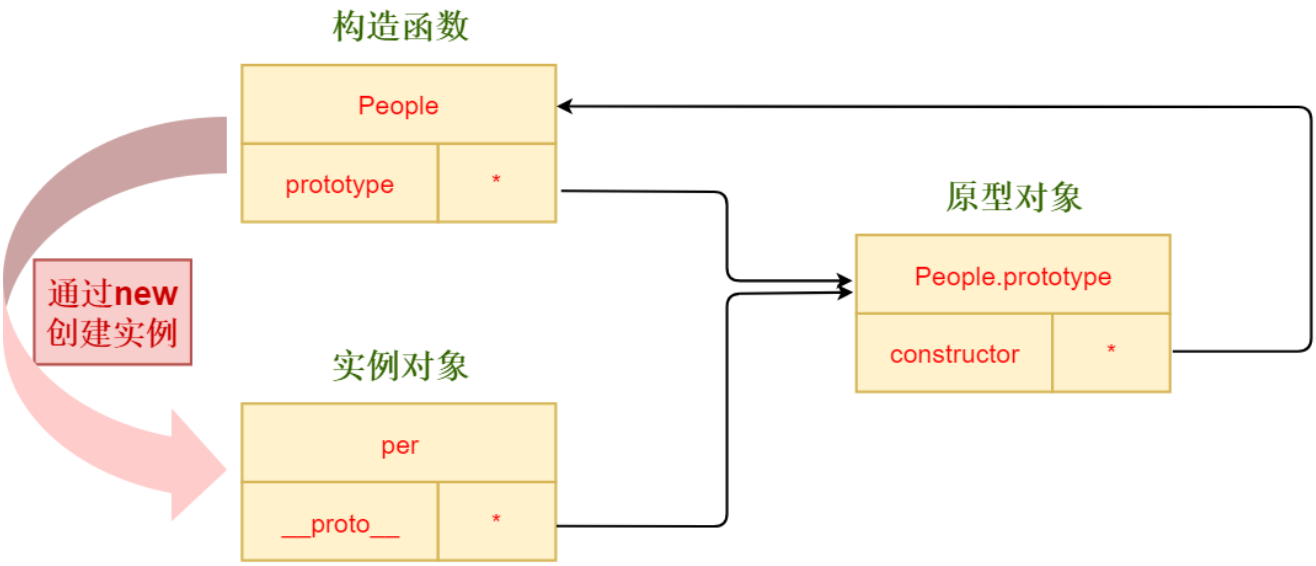
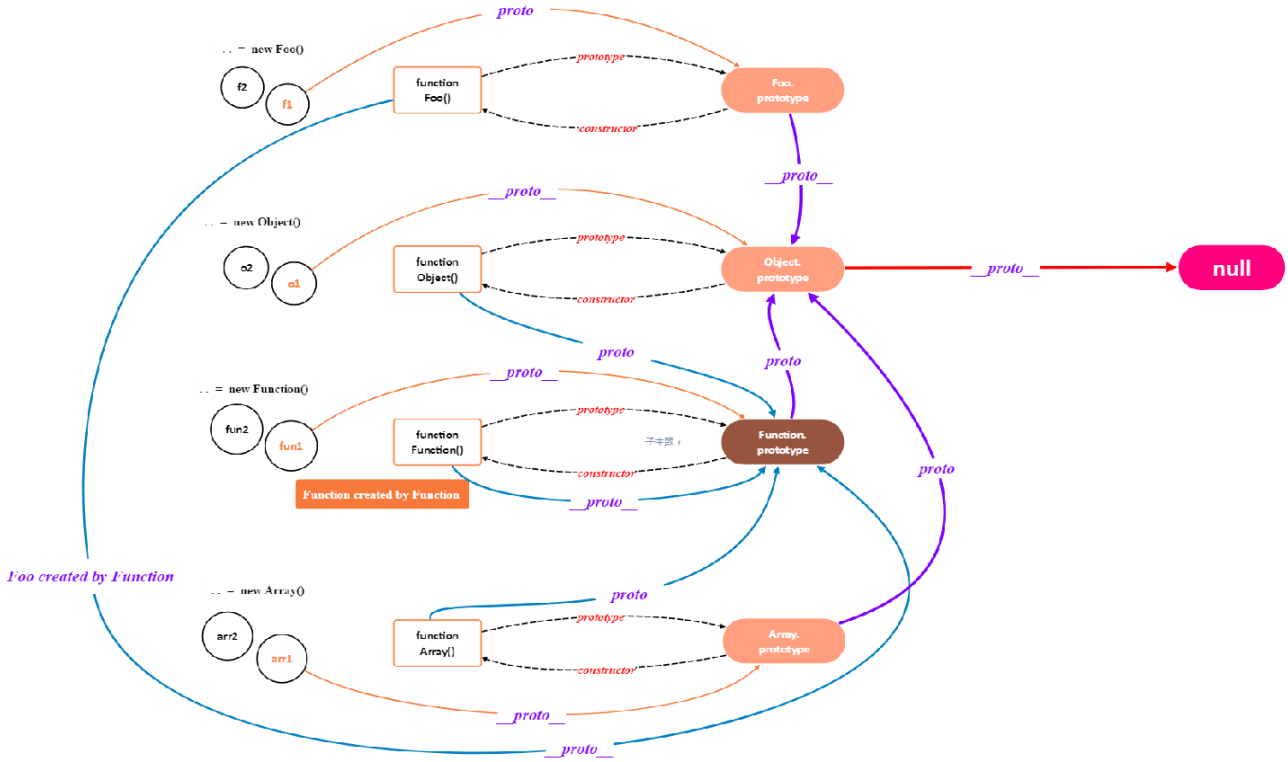
原型链

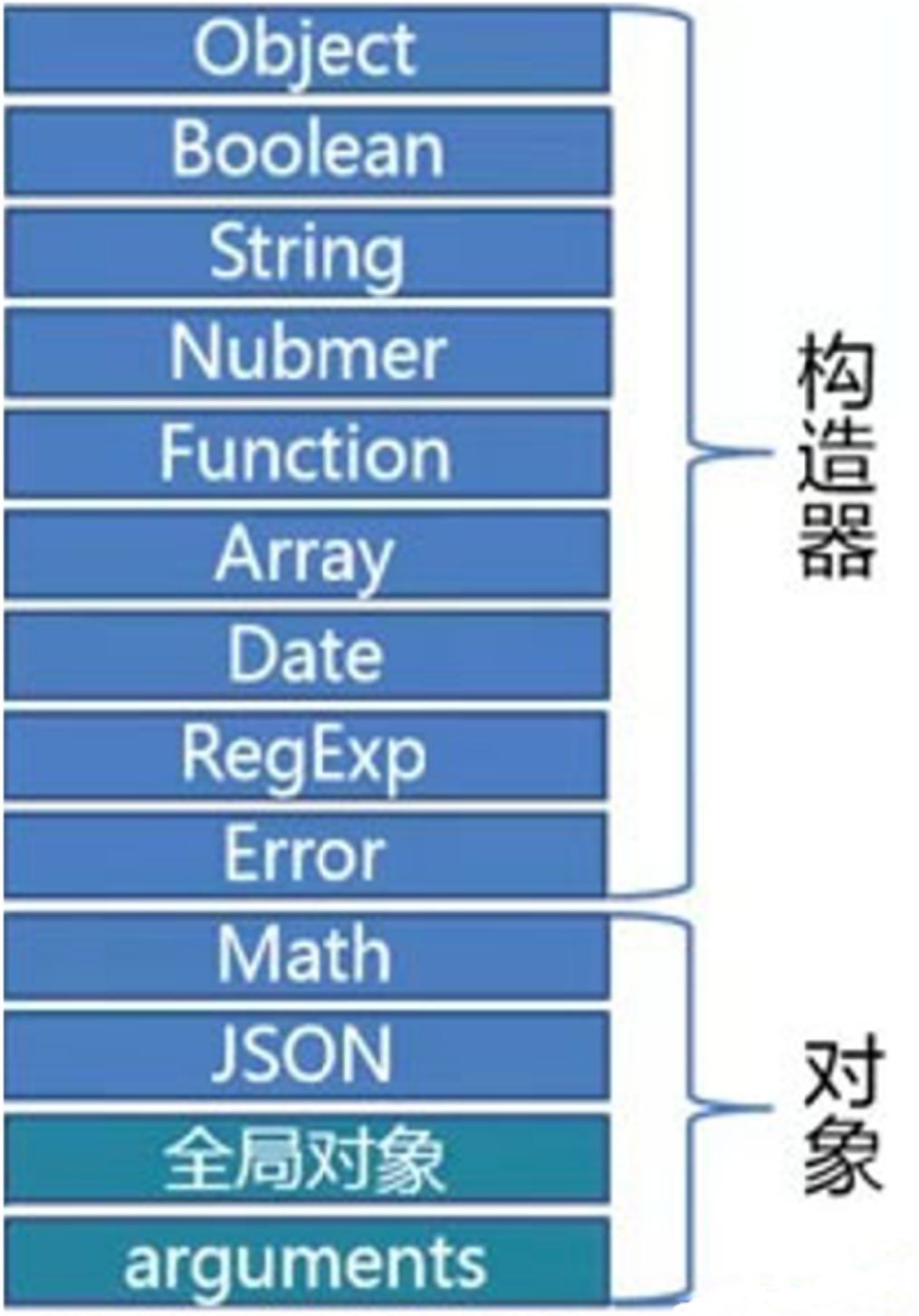
原型



原型链

1. 每个对象拥有一个原型对象，对象以其原型为模板、从原型继承方法和属性。原型对象也可能拥有原型，并从中继承方法和属性，一层一层、以此类推。这种关系常被称为原型链
2. 每个实例对象（object ）都有一个属性（**proto**）指向它的原型对象（prototype）。该原型对象也有一个自己的原型对象(**proto**)，层层向上直到一个对象的原型对象为 null。根据定义，null 没有原型，并作为这个原型链中的最后一个环节。
3. 所有对象具有 **proto** 属性。
4. 函数具有 prototype 属性，Function.prototype 是例外，虽然是函数，但是不具有 prototype 属性。
5. 函数具有prototype、**proto**属性。
6. Function.prototype.**proto** == Object.prototype 给 Object.prototype 添加方法和属性，Function.prototype 也会拥有相同的方法和属性，相反则不具有
7. Function.prototype 是函数的起源，Object.prototype 是对象的起源
8. 函数也是对象，是对象中的一等公民。


. . = new Foo()
f2
f1
function Foo()
prototype
constructor
Foo. prototype
__proto__
proto
. . = new Object()
o2
o1
function Object()
Object. prototype
null
. . = new Function()
fun2
fun1
function Function()
Function. prototype
Function created by Function
Foo created by Function
. . = new Array()
arr2
arr1
function Array()
Array. prototype



获取/设置/判断原型对象

1. Object.getPrototypeOf(obj) 返回指定对象的原型，如果没有继承属性，则返回 null 。
2. Object.create(proto) 创建一个新对象，使用现有的对象来提供新创建的对象的**proto**。
3. prototypeObj.isPrototypeOf(object) 测试 prototypeObj 对象是否存在 object 对象的原型链上。