

Persistence-via Mapping Object



panscan.ivow.bet/task/viewdemo%2Fdemo_5_3_state_save#/



input: 0

0-100



[[save](#)]

```
obj_str = {"json_class":"StateSave","data":{"\n":0}}
```

```
obj_encode_str = eJyrVsoqzs+LT85JLC5WsIIKLkksSQ1OLEtV0IFKSSxJBAPVxyjlxShZGdQq 1QIAZ24Ozg==
```

Task Params

obj_str = eJyrVsoqzs+LT85JLC5v

```
class StateSave < MappingObject
  mapping_accessor :n

  def set(n)
    self.n = n
  end

  def get
    self.n
  end
end

def main()

  obj_str = '__obj_str__'

  begin
    state = MappingObject.from_encode_str(obj_str)
  rescue =>e
    $logger.call "error #{e}"
    state = StateSave.new
  end
end
```

- Mapping Object - 连通对象，前后端桥梁
- 前端的对象，进行序列化+Base64编码+压缩，变成编码字符串，存到任务参数变量
- 然后在后端执行这个任务，把编码字符串参数写入HTML中
- 下次浏览器打开，对编码字符串进行解压+Base64展开+反序列化，在前端中构建对象实例

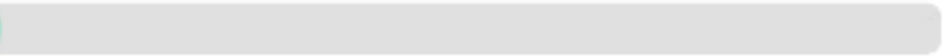


panscan.ivow.bet/task/viewdemo%2Fdemo_5_3_state_save#/



input: 0

0-100



[[save](#)]

```
obj_str = {"json_class":"StateSave","data":{"\n":0}}
```

```
obj_encode_str = eJyrVsoqzs+LT85JLC5WsIIKLkksSQ1OLEtV0IFKSSxJBAPVxyjlxShZGdQq 1QIAZ24Ozg==
```



panscan.ivow.bet/task/view/demo%2Fdemo_5_3_state_save#/



input: 0

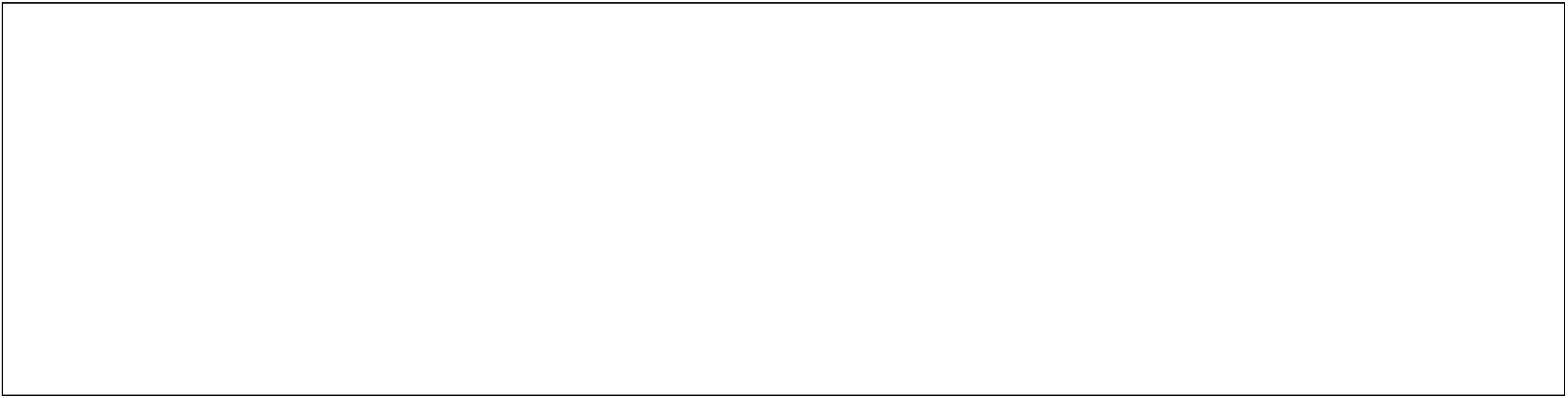
0-100



[[save](#)]

```
obj_str = {"json_class":"StateSave","data":{"\n":0}}
```

```
obj_encode_str = eJyrVsoqzs+LT85JLC5WsIIKLkksSQ1OLEtV0IFKSSxJBAPVxyjlxShZGdQq 1QIAZ24Ozg==
```

Persistence - via Mapping Object

Task Params

obj_str = eJyrVsoqzs+LT85JLC5v

```
class StateSave < MappingObject
  mapping_accessor :n

  def set(n)
    self.n = n
  end

  def get
    self.n
  end
end

def main()


  obj_str = '__obj_str__'

  begin
    state = MappingObject.from_encode_str(obj_str)
  rescue =>e
    $logger.call "error #{e}"
    state = StateSave.new
  end
end
```

- Mapping Object - 连通对象，前后端桥梁
- 前端的对象，进行序列化+Base64编码+压缩，变成编码字符串，存到任务参数变量
- 然后在后端执行这个任务，把编码字符串参数写入HTML中
- 下次浏览器打开，对编码字符串进行解压+Base64展开+反序列化，在前端中构建对象实例

← → ↻ 🔒 panscan.ivow.bet/task/view/demo%2Fdemo_5_3_state_save#/

input: 0

0-100  [[save](#)]

obj_str = {"json_class":"StateSave","data":{"\n\:0"}}
obj_encode_str = eJyrVsoqzs+LT85JLC5WslIKLkksSQ1OLEtV0IFKSSxJBAPVxyjlxShZGdQq 1QIAZ24Ozg==

Trick - Return CPU to UI Thread

- 在前端Runtime做耗时的任务，会阻碍UI的响应，整个页面卡死
- 需要把控制权交还给UI Thread
 - `(1..100).each {|x| many very heavy calculation }`
- `==>`
 - `def single_calc(x)`
 - `one very heavy calculation`
 - `setTimeout(->() { single_calc(x+1) }, 1)`
 - `end`