

# 《JavaScript编程全解》

## 图书基本信息

书名：《JavaScript编程全解》

13位ISBN编号：9787115333414

10位ISBN编号：7115333416

出版时间：2013-12

出版社：人民邮电出版社

作者：[日]井上诚一郎, [日]土江拓郎, [日]滨边将太

页数：420

译者：陈筱烟

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：[www.tushu111.com](http://www.tushu111.com)

# 《JavaScript编程全解》

## 内容概要

本书全方位地介绍了JavaScript开发中的各个主题，无论是前端还是后端的JavaScript开发者都可以在本书中找到自己需要的内容。本书对HTML5、Web API、Node.js及WebSocket等最新的热门技术也作了深入浅出的介绍，并提供了大量实际应用范例。

本书语法说明系统深入、示例代码规范详细，对容易产生问题之处均做了重点说明，不仅适合初学者入门，而且有经验的JavaScript开发人员、项目负责人也能从中受益。

## 作者简介

作者简介：

井上诚一郎

曾在美国参与过Lotus Notes的开发，后在日本创立了Ariel Network股份公司，任CTO。目前从事面向企业的PSP软件及企业产品的开发。著有《PSP教科书》、《Java编程详解》、《实践JS 服务器端JavaScript入门》等书。负责本书Part1、Part2、Part5与Part6的撰写。

土江拓郎

大学时学习了航天工程学和机器人工程学，之后凭着兴趣进入了IT行业工作。2008年加入Ariel Network股份公司。从事Java及JavaScript相关的企业产品开发工作。负责本书Part3的撰写。

滨边将太

学生时代在Ariel Network股份公司实习。学习了软件开发的基础知识并了解了开发人员的工作生活情况。2009年加入了雅虎公司，从事针对电视的软键盘开发，以及智能手机应用GyaO!的开发。最近正在公司中开展HTML5及Node.js的普及活动。负责了本书Part4的撰写。

译者简介：

陈筱烟

毕业于复旦大学计算机科学与技术系，主要研究方向为跨设备人机交互理论。长期从事对日软件外包工作。从大学时期开始接触并使用Java、JavaScript进行程序开发，现在对Web应用及智能手机应用的开发很感兴趣。

## 书籍目录

第1部分 JavaScript概要	001
第1章 JavaScript概要	002
1.1 JavaScript概要	002
1.2 JavaScript的历史	002
1.3 ECMAScript	003
1.3.1 JavaScript的标准化	003
1.3.2 被放弃的ECMAScript第4版	004
1.4 JavaScript的版本	004
1.5 JavaScript实现方式	005
1.6 JavaScript运行环境	006
1.6.1 核心语言	006
1.6.2 宿主对象	007
1.7 JavaScript相关环境	007
1.7.1 库	007
1.7.2 源代码压缩	007
1.7.3 集成开发环境	008
第2部分 JavaScript的语言基础	009
第2章 JavaScript基础	010
2.1 JavaScript的特点	010
2.2 关于编排格式	011
2.3 变量的基础	012
2.3.1 变量的使用方法	012
2.3.2 省略var	013
2.3.3 常量	013
2.4 函数基础	013
2.4.1 函数的定义	013
2.4.2 函数的声明与调用	014
2.4.3 匿名函数	014
2.4.4 函数是一种对象	015
2.5 对象的基础	016
2.5.1 对象的定义	016
2.5.2 对象字面量表达式与对象的使用	016
2.5.3 属性访问	017
2.5.4 属性访问（括号方式）	017
2.5.5 方法	018
2.5.6 new表达式	018
2.5.7 类与实例	018
2.5.8 对类的功能的整理	018
2.5.9 对象与类型	019
2.6 数组的基础	019
第3章 JavaScript的数据类型	021
3.1 数据类型的定义	021
3.1.1 在数据类型方面与Java作比较	021
3.1.2 基本数据类型和引用类型	022
3.2 内建数据类型概要	022
3.3 字符串型	023
3.3.1 字符串字面量	023

3.3.2	字符串型的运算	024
3.3.3	字符串型的比较	024
3.3.4	字符串类 (String类)	025
3.3.5	字符串对象	026
3.3.6	避免混用字符串值和字符串对象	027
3.3.7	调用String函数	027
3.3.8	String类的功能	027
3.3.9	非破坏性的方法	029
3.4	数值型	029
3.4.1	数值字面量	029
3.4.2	数值型的运算	030
3.4.3	有关浮点数的常见注意事项	030
3.4.4	数值类 (Number类)	031
3.4.5	调用Number函数	031
3.4.6	Number类的功能	032
3.4.7	边界值与特殊数值	033
3.4.8	NaN	034
3.5	布尔型	035
3.5.1	布尔值	035
3.5.2	布尔类 (Boolean类)	036
3.5.3	Boolean类的功能	036
3.6	null型	037
3.7	undefined型	037
3.8	Object类型	038
3.9	数据类型转换	039
3.9.1	从字符串值转换为数值	039
3.9.2	从数值转换为字符串值	040
3.9.3	数据类型转换的惯用方法	040
3.9.4	转换为布尔型	041
3.9.5	其他的数据类型转换	042
3.9.6	从Object类型转换为基本数据类型	042
3.9.7	从基本数据类型转换为Object类型	043
第4章	语句、表达式和运算符	045
4.1	表达式和语句的构成	045
4.2	保留字	045
4.3	标识符	046
4.4	字面量	047
4.5	语句	047
4.6	代码块 (复合语句)	048
4.7	变量声明语句	048
4.8	函数声明语句	048
4.9	表达式语句	048
4.10	空语句	049
4.11	控制语句	049
4.12	if-else语句	050
4.13	switch-case语句	052
4.14	循环语句	054
4.15	while语句	055
4.16	do-while语句	056

4.17	for语句	057
4.18	for in语句	058
4.18.1	数列与for in语句	059
4.18.2	在使用for in语句时需要注意的地方	060
4.19	for each in语句	060
4.20	break语句	061
4.21	continue语句	061
4.22	通过标签跳转	062
4.23	return语句	063
4.24	异常	063
4.25	其他	064
4.26	注释	065
4.27	表达式	065
4.28	运算符	065
4.29	表达式求值	066
4.30	运算符的优先级以及结合律	066
4.31	算术运算符	067
4.32	字符串连接运算符	068
4.33	相等运算符	068
4.34	比较运算符	069
4.35	in运算符	070
4.36	instanceof运算符	071
4.37	逻辑运算符	071
4.38	位运算符	072
4.39	赋值运算符	072
4.40	算术赋值运算符	073
4.41	条件运算符（三目运算符）	073
4.42	typeof运算符	073
4.43	new运算符	074
4.44	delete运算符	074
4.45	void运算符	074
4.46	逗号(,)运算符	074
4.47	点运算符和中括号运算符	075
4.48	函数调用运算符	075
4.49	运算符使用以及数据类型转换中需要注意的地方	075
第5章	变量与对象	076
5.1	变量的声明	076
5.2	变量与引用	076
5.2.1	函数的参数（值的传递）	078
5.2.2	字符串与引用	079
5.2.3	对象与引用相关的术语总结	079
5.3	变量与属性	080
5.4	变量的查找	081
5.5	对变量是否存在的检验	081
5.6	对象的定义	082
5.6.1	抽象数据类型与面向对象	082
5.6.2	实例间的协作关系与面向对象	083
5.6.3	JavaScript的对象	083
5.7	对象的生成	083

5.7.1	对象字面量	083
5.7.2	构造函数与new表达式	085
5.7.3	构造函数与类的定义	087
5.8	属性的访问	087
5.8.1	属性值的更新	088
5.8.2	点运算符与中括号运算符在使用上的区别	088
5.8.3	属性的枚举	089
5.9	作为关联数组的对象	089
5.9.1	关联数组	089
5.9.2	作为关联数组的对象的注意点	090
5.10	属性的属性	091
5.11	垃圾回收	092
5.12	不可变对象	092
5.12.1	不可变对象的定义	092
5.12.2	不可变对象的作用	092
5.12.3	实现不可变对象的方式	093
5.13	方法	094
5.14	this引用	094
5.14.1	this引用的规则	094
5.14.2	this引用的注意点	095
5.15	apply与call	096
5.16	原型继承	097
5.16.1	原型链	097
5.16.2	原型链的具体示例	099
5.16.3	原型继承与类	100
5.16.4	对于原型链的常见误解以及_proto_属性	100
5.16.5	原型对象	101
5.16.6	ECMAScript第5版与原型对象	101
5.17	对象与数据类型	102
5.17.1	数据类型判定 (constructor属性)	102
5.17.2	constructor属性的注意点	102
5.17.3	数据类型判定 (instance运算与isPrototypeOf方法)	103
5.17.4	数据类型判定 (鸭子类型)	103
5.17.5	属性的枚举 (原型继承的相关问题)	104
5.18	ECMAScript第5版中的Object类	105
5.18.1	属性对象	105
5.18.2	访问器的属性	106
5.19	标准对象	108
5.20	Object类	108
5.21	全局对象	110
5.21.1	全局对象与全局变量	110
5.21.2	Math对象	111
5.21.3	Error对象	112
第6章	函数与闭包	113
6.1	函数声明语句与匿名函数表达式	113
6.2	函数调用的分类	113
6.3	参数与局部变量	114
6.3.1	arguments对象	114
6.3.2	递归函数	114

6.4	作用域	115
6.4.1	浏览器与作用域	116
6.4.2	块级作用域	116
6.4.3	let与块级作用域	117
6.4.4	嵌套函数与作用域	119
6.4.5	变量隐藏	119
6.5	函数是一种对象	120
6.6	Function类	122
6.7	嵌套函数声明与闭包	123
6.7.1	对闭包的初步认识	123
6.7.2	闭包的原理	123
6.7.3	闭包中需要注意的地方	126
6.7.4	防范命名空间的污染	127
6.7.5	闭包与类	129
6.8	回调函数设计模式	130
6.8.1	回调函数与控制反转	130
6.8.2	JavaScript与回调函数	131
第7章	数据处理	134
7.1	数组	134
7.1.1	JavaScript的数组	134
7.1.2	数组元素的访问	135
7.1.3	数组的长度	136
7.1.4	数组元素的枚举	136
7.1.5	多维数组	137
7.1.6	数组是一种对象	138
7.1.7	Array类	139
7.1.8	数组对象的意义	140
7.1.9	数组的习惯用法	141
7.1.10	数组的内部实现	144
7.1.11	数组风格的对象	145
7.1.12	迭代器	145
7.1.13	生成器	147
7.1.14	数组的内包	149
7.2	JSON	149
7.2.1	JSON字符串	149
7.2.2	JSON对象	150
7.3	日期处理	151
7.4	正则表达式	153
7.4.1	正则表达式的定义	153
7.4.2	正则表达式相关的术语	154
7.4.3	正则表达式的语法	154
7.4.4	JavaScript中的正则表达式	156
7.4.5	正则表达式程序设计	157
7.4.6	字符串对象与正则表达式对象	158
第3部分	客户端JavaScript	161
第8章	客户端JavaScript与HTML	162
8.1	客户端JavaScript的重要性	162
8.1.1	Web应用程序的发展	162
8.1.2	JavaScript的性能提升	162



8.1.3	JavaScript的作用	163
8.2	HTML与JavaScript	163
8.2.1	网页显示过程中的处理流程	163
8.2.2	JavaScript的表述方式及其执行流程	163
8.2.3	执行流程的小结	166
8.3	运行环境与开发环境	166
8.3.1	运行环境	166
8.3.2	开发环境	166
8.4	调试	167
8.4.1	alert	167
8.4.2	console	167
8.4.3	onerror	169
8.4.4	Firebug, Web Inspector (Developer Tools), Opera Dragonfly	169
8.5	跨浏览器支持	171
8.5.1	应当提供支持的浏览器	171
8.5.2	实现方法	172
8.6	Window对象	174
8.6.1	Navigator对象	174
8.6.2	Location对象	174
8.6.3	History对象	175
8.6.4	Screen对象	176
8.6.5	对Window对象的引用	176
8.6.6	Document对象	176
第9章	DOM	177
9.1	DOM的定义	177
9.1.1	DOM Level 1	177
9.1.2	DOM Level 2	177
9.1.3	DOM Level 3	178
9.1.4	DOM的表述方式	178
9.2	DOM的基础	179
9.2.1	标签、元素、节点	179
9.2.2	DOM操作	179
9.2.3	Document对象	179
9.3	节点的选择	180
9.3.1	通过ID检索	180
9.3.2	通过标签名检索	180
9.3.3	通过名称检索	184
9.3.4	通过类名检索	184
9.3.5	父节点、子节点、兄弟节点	185
9.3.6	XPath	187
9.3.7	Selector API	189
9.4	节点的创建与新增	190
9.5	节点的内容更改	190
9.6	节点的删除	190
9.7	innerHTML/textContent	190
9.7.1	innerHTML	190
9.7.2	textContent	191
9.8	DOM操作的性能	191
第10章	事件	192

10.1	事件驱动程序设计	192
10.2	事件处理程序/事件侦听器的设定	192
10.2.1	指定为HTML元素的属性	193
10.2.2	指定为DOM元素的属性	194
10.2.3	通过EventTarget.addEventListener()进行指定	194
10.2.4	事件处理程序/事件侦听器内的this引用	196
10.3	事件的触发	196
10.4	事件的传播	196
10.4.1	捕获阶段	197
10.4.2	目标阶段	197
10.4.3	事件冒泡阶段	197
10.4.4	取消	197
10.5	事件所具有的元素	198
10.6	标准事件	199
10.6.1	DOM Level 2中所定义的事件	199
10.6.2	DOM Level 3中所定义的事件	200
10.7	自定义事件	202
第11章	客户端JavaScript实践	203
11.1	样式	203
11.1.1	样式的变更方法	203
11.1.2	位置的设定	207
11.1.3	位置	208
11.1.4	动画	209
11.2	AJAX	210
11.2.1	异步处理的优点	210
11.2.2	XMLHttpRequest	210
11.2.3	基本的处理流程	210
11.2.4	同步通信	212
11.2.5	超时	212
11.2.6	响应	213
11.2.7	跨源限制	214
11.2.8	跨源通信	214
11.2.9	JSONP	214
11.2.10	iframe攻击 (iframe hack)	215
11.2.11	window.postMessage	218
11.2.12	XMLHttpRequest Level 2	219
11.2.13	跨源通信的安全问题	219
11.3	表单	219
11.3.1	表单元素	219
11.3.2	表单控件	221
11.3.3	内容验证	221
11.3.4	可用于验证的事件	222
11.3.5	使用表单而不产生页面跳转的方法	222
第12章	库	224
12.1	使用库的原因	224
12.2	jQuery的特征	224
12.3	jQuery的基本概念	225
12.3.1	使用实例	225
12.3.2	链式语法	226

12.4	\$函数	227
12.4.1	抽取与选择器相匹配的元素	227
12.4.2	创建新的DOM元素	227
12.4.3	将已有的DOM元素转换为jQuery对象	227
12.4.4	对DOM构造完成后的事件侦听器进行设定	227
12.5	通过jQuery进行DOM操作	228
12.5.1	元素的选择	228
12.5.2	元素的创建·添加·替换·删除	230
12.6	通过jQuery处理事件	231
12.6.1	事件侦听器的注册·删除	231
12.6.2	事件专用的事件侦听器注册方法	232
12.6.3	ready()方法	232
12.7	通过jQuery对样式进行操作	233
12.7.1	基本的样式操作	233
12.7.2	动画	234
12.8	通过jQuery进行AJAX操作	235
12.8.1	AJAX()函数	235
12.8.2	AJAX()的包装函数	236
12.8.3	全局事件	237
12.9	Deferred	237
12.9.1	Deferred的基本概念	237
12.9.2	状态迁移	238
12.9.3	后续函数	239
12.9.4	并行处理	241
12.10	jQuery插件	241
12.10.1	使用jQuery插件	241
12.10.2	创建jQuery插件	242
12.11	与其他库共同使用	243
12.11.1	\$对象的冲突	243
12.11.2	避免\$对象的冲突	243
12.12	库的使用方法	244
第13章	HTML5概要	272
13.1	HTML5的历史	246
13.2	HTML5的现状	247
13.2.1	浏览器的支持情况	247
13.2.2	Web应用程序与原生应用程序	248
13.3	HTML5的概要	248
第14章	Web应用程序	250
14.1	History API	250
14.1.1	History API的定义	250
14.1.2	哈希片段	250
14.1.3	接口	251
14.2	ApplicationCache	255
14.2.1	关于缓存管理	255
14.2.2	缓存清单文件	255
14.2.3	ApplicationCache API	258
14.2.4	在线与离线	259
第15章	与桌面应用的协作	260
15.1	Drag Drop API	260

15.1.1	Drag Drop API的定义	260
HTML5		245
第4部分	HTML5	245
15.1.2	接口	261
15.1.3	基本的拖动与释放	262
15.1.4	自定义显示	263
15.1.5	文件的Drag-In/ Drag-Out	265
15.2	File API	267
15.2.1	File API的定义	267
15.2.2	File对象	267
15.2.3	FileReader	269
15.2.4	data URL	271
15.2.5	FileReaderSync	273
第16章	存储	274
16.1	Web Storage	274
16.1.1	Web Storage的定义	274
16.1.2	基本操作	275
16.1.3	storage事件	277
16.1.4	关于Cookie	277
16.1.5	命名空间的管理	278
16.1.6	版本的管理	279
16.1.7	对localStorage的模拟	279
16.2	Indexed Database	280
16.2.1	Indexed Database的定义	280
16.2.2	基础架构	280
16.2.3	连接数据库	281
16.2.4	对象存储的创建	281
16.2.5	数据的添加·删除·引用	282
16.2.6	索引的创建	283
16.2.7	数据的检索与更新	284
16.2.8	数据的排序	285
16.2.9	事务	285
16.2.10	同步API	286
第17章	WebSocket	287
17.1	WebSocket概要	287
17.1.1	WebSocket的定义	287
17.1.2	现有的通信技术	287
17.1.3	WebSocket的标准	290
17.1.4	WebSocket的执行方式	290
17.2	基本操作	291
17.2.1	连接的建立	291
17.2.2	消息的收发	291
17.2.3	连接的切断	292
17.2.4	连接的状态确认	292
17.2.5	二进制数据的收发	293
17.2.6	WebSocket实例的属性一览	293
17.3	WebSocket实践	294
17.3.1	Node.js的安装	294
17.3.2	服务器端的实现	295

17.3.3	客户端的实现	295
17.3.4	客户端的实现2	296
第18章	Web Workers	298
18.1	Web Workers概要	298
18.1.1	Web Workers的定义	298
18.1.2	Web Workers的执行方式	298
18.2	基本操作	299
18.2.1	工作线程的创建	299
18.2.2	主线程一侧的消息收发	299
18.2.3	工作线程一侧的消息收发	300
18.2.4	工作线程的删除	300
18.2.5	外部文件的读取	301
18.3	Web Worker实践	301
18.3.1	工作线程的使用	301
18.3.2	中断对工作线程的处理	302
18.4	共享工作线程	304
18.4.1	共享工作线程的定义	304
18.4.2	共享工作线程的创建	304
18.4.3	共享工作线程的消息收发	305
18.4.4	共享工作线程的删除	306
18.4.5	共享工作线程的应用实例	306
第5部分	Web API	309
第19章	Web API的基础	310
19.1	Web API与Web服务	310
19.2	Web API的历史	311
19.2.1	Web抓取	311
19.2.2	语义网	311
19.2.3	XML	311
19.2.4	Atom	312
19.2.5	JSON	312
19.2.6	SOAP	313
19.2.7	REST	313
19.2.8	简单总结	313
19.3	Web API的组成	314
19.3.1	Web API的形式	314
19.3.2	Web API的使用	315
19.3.3	RESTful API	315
19.3.4	API密钥	316
19.4	用户验证与授权	317
19.4.1	Web应用程序的会话管理	317
19.4.2	会话管理与用户验证	318
19.4.3	Web API与权限	319
19.4.4	验证与授权	320
19.4.5	OAuth	321
第20章	Web API的实例	323
20.1	Web API的分类	323
20.2	Google Translate API	324
20.2.1	准备	325
20.2.2	执行方式的概要	325

20.2.3	使用了Web API的代码示例	326
20.2.4	微件 ( Google Translate Element )	327
20.3	Google Maps API	328
20.3.1	Google Static Maps API	328
20.3.2	我的地图	329
20.3.3	Google Maps API的概要	330
20.3.4	简单的Google Maps API示例	330
20.3.5	事件	331
20.3.6	Geolocation API与Geocoding API	333
20.4	Yahoo! Flickr	334
20.4.1	Flickr Web API的使用	335
20.4.2	Flickr Web API的使用实例	336
20.5	Twitter	337
20.5.1	搜索API	337
20.5.2	REST API	338
20.5.3	Twitter JS API @anywhere	339
20.5.4	Twitter Widget	341
20.6	Facebook	341
20.6.1	Facebook应用的发展历程	341
20.6.2	Facebook的JavaScript API	343
20.6.3	Facebook的插件	344
20.7	OpenSocial	345
第6部分	服务器端 JavaScript	351
第21章	服务器端JavaScript与Node.js	352
21.1	服务器端JavaScript的动向	352
21.2	CommonJS	352
21.2.1	CommonJS的定义	352
21.2.2	CommonJS的动向	353
21.2.3	模块功能	353
21.3	Node.js	355
21.3.1	Node.js概要	355
21.3.2	node指令	359
21.3.3	npm与包	359
21.3.4	console模块	360
21.3.5	util模块	361
21.3.6	process对象	362
21.3.7	全局对象	363
21.3.8	Node.js程序设计概要	363
21.3.9	事件API	365
21.3.10	缓冲	369
21.3.11	流	372
第22章	Node.js程序设计实践	374
22.1	HTTP服务器处理	374
22.1.1	HTTP服务器处理的基本流程	374
22.1.2	请求处理	375
22.1.3	响应处理	376
22.1.4	POST请求处理	377
22.2	HTTP客户端处理	378
22.3	HTTPS处理	379

22.3.1	通过openssl指令发布自签名证书的方法	379
22.3.2	HTTPS服务器	379
22.4	Socket.IO与WebSocket	380
22.5	下层网络程序设计	381
22.5.1	下层网络处理	381
22.5.2	套接字的定义	382
22.5.3	套接字程序设计的基本结构	382
22.5.4	套接字程序设计的具体实例	384
22.6	文件处理	385
22.6.1	本节的范例代码	385
22.6.2	文件的异步处理	386
22.6.3	文件的同步处理	386
22.6.4	文件操作相关函数	387
22.6.5	文件读取	387
22.6.6	文件写入	388
22.6.7	目录操作	389
22.6.8	对文件更改的监视	390
22.6.9	文件路径	390
22.7	定时器	390
22.8	Express	391
22.8.1	URL路由	392
22.8.2	请求处理	392
22.8.3	响应处理	393
22.8.4	scaffold创建功能	393
22.8.5	MVC架构	393
22.8.6	模板语言Jade	394
22.8.7	MongoDB (数据库)	395
22.8.8	Mongoose的实例	397
22.8.9	使用了Express与Mongoose的Web应用程序	398
后记		401
索引		403

## 精彩短评

- 1、很细，有些地方不深入
- 2、还行
- 3、内容很全的JS学习书，把HTML5都涵盖了。
- 4、全面
- 5、没耐心的人千万别读这本书，太细了
- 6、比犀牛书好啃。
- 7、这本书的内容很杂很广，适合想了解Javascript的程序员读，适合JS入门。
- 8、不推荐拿来入门，还是选《JavaScript高级程序设计》与《JavaScript权威指南》吧
- 9、正在读，一直有所收获。作者会谈一些实际开发的经验和忠告。
- 10、日本人一贯的技术书风格，全面娓娓道来但点到即止。
- 11、日本人写的教材，非常的细致，由简入繁，而且覆盖面广博，非常不错的入门教材
- 12、讲解细致，可以案头的参考书。
- 13、两天时间看完，收获不小，各种细节的讲解很到位
- 14、很细致但是不适合入门
- 15、看完了，所有涉及js的知识点都有提到，非常赞！建议看。



## 精彩书评

1、看了几本日本人写过的技术书，我觉得是大爱，基本上是本本出精品。前面几章的基础知识，虽然我熟的不能再熟，但是看起来一点也不会让我烦躁。里面所有的知识点都细致的讲出来，让人觉得这本书很严谨，但又不枯燥。对比于国人写的技术书，很多都是全章Copy代码，不严谨，不专业。

## 章节试读

### 1、《JavaScript编程全解》的笔记-第109页

在没有特别理由的情况下，建议通过字面量来生成对象，而不要使用Object类的函数或构造函数调用。

### 2、《JavaScript编程全解》的笔记-第111页

表5.9中 isFinite(num)的说明出现错误：真假颠倒了

## 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:[www.tushu111.com](http://www.tushu111.com)