# 测试项目符号3

## 目录

API测试是一种软件测试，用于验证应用编程接口(API)以确保其满足对功能、可靠性、性能和安全性的期望。它主要关注业务逻辑层和软件系统之间的数据交换，与用户界面(UI)无关

这是为了测试文本之间的URL

https://policycentral.cloudapps.cisco.com/cppc/policy-advisor/policies/view-policy/1624

思科的业务行为准则(COBC)反映了我们如何诚信地工作和做出决策。它还提供资源，帮助解决复杂问题，如负责任的人工智能使用和利益冲突。

```
function reverseString(str) {
  return str.split("").reverse().join("");
}
```

https://cisco.account.box.com/login

https://cisco.service-now.com/helpzone?id=sc_cat_item&sys_id=a9860b89dbd9a640cb5772fc0f96191d&u_business_service=

请求与您遇到的问题相关的帮助。将创建和管理事件记录，直至成功解决问题。您还将收到有关进度的通知。

https://www.geeksforgeeks.org/software-testing/software-testing-manual-testing/

在Black Box Testing技术中，测试人员或QA分析人员仅通过手动提供不同的测试案例来检查特定模块或特定方法的功能，有时也检查整个应用程序的功能。在这里，测试人员将给出应用程序的输入并手动对其进行测试。

如果返回预期输出，测试人员将继续进行另一组输入并将所有结果报告给团队。如果用户在测试期间手动提供的输入失败，他/她将向开发团队报告此问题。

测试视频

| 检查 | 表 |
| --- | --- |

| | 检查链接 |
|---|---|

测试表

https://cisco.service-now.com/now/sow/record/incident/507c393193e672502c66ff60ed03d632

## 白盒测试

在白盒测试技术中，人员需要手动检查系统的内部结构，如设计、编码等。在这里，开发团队将逐行检查整个编码部件以确保代码的正确性。

如果他/她在代码中发现任何差异或错误，他们将会纠正或修复编码或设计中的错误。在这里，该过程完全由人工执行，并且由于该检查代码或设计由人工手动检查，该过程是高效的。

https://en.wikipedia.org/wiki/Manual_testing

https://www.google.com/

“bdb developer role”检查已从ART API迁移到Entra ID within One Access。请求访问时，请确保选择“集成方法：memberOf",因为有两个权利具有相同的名称。

API测试的主要方面 

- 消息层通信:API测试不是使用GUI，而是使用各种HTTP方法(GET、POST、PUT、DELETE)以及数据格式（如JSON或XML）直接与应用程序的终端(URI)交互。
- 早期缺陷检测:API测试可以在软件开发生命周期早期执行，甚至可以在构建UI之前执行，使团队能够以更低的成本更高效地查找和修复问题。
- 自动化重点:由于直接交互的本质和一致结构，API测试非常适合自动化，这一点在现代敏捷和DevOps环境中对于在CI/CD管道中连续测试至关重要。
- 全面覆盖:与单独的UI测试相比，它提供的测试覆盖范围更广，包括测试边缘案例、错误处理以及可能难以通过UI访问的安全漏洞。

API测试类型 
使用不同类型的API测试来涵盖应用程序质量的各个方面： 

卡塔隆

- 功能测试:验证API是否正确执行其预期操作，按照指定处理输入、输出和状态代码。
- 性能测试:评估API在各种负载条件（例如峰值流量、应力）下的速度、稳定性和可扩展性。
- 安全测试:识别漏洞，例如SQL注入、跨站点脚本(XSS)和中断的身份验证/授权，以保护敏感数据。
- 集成测试:确认API交互的系统不同部分或外部服务无缝地协同工作。
- 合同测试:确保API遵守协定的合同（规范，如OpenAPI/Swaggeror WSDL），防止在服务更新之间中断更改。

- 端到端测试:验证涉及链接在一起的多个API调用的整个用户过程。
- API测试类型 □
  使用不同类型的API测试来涵盖应用程序质量的各个方面： □

  手动测试的第一步是了解软件的预期功能。

  - 功能要求:验证功能，例如正确的用户登录。
  - 非功能要求：验证性能、可用性和安全性（例如，页面加载时间少于2秒）。
  - 用户案例和设计文档:了解用户交互和工作流程。
  - 利益相关方投入：向客户、产品经理或设计人员明确要求。

### 步骤 2：创建测试计划

测试计划定义了测试策略和目标。

- 范围：标识要测试的功能和排除项。
- 目标：确保核心功能和用户体验。
- 资源：指定团队成员、工具和时间表。
- 测试技术：包括功能、可用性和探索性测试。
- 环境：定义分段或类似生产的设置。

### 步骤 3：设计测试案例

测试案例是清晰的逐步脚本，可确保全面的手动测试。测试用例充当测试者的详细指南，确保对所有场景都进行检查。每个测试案例包括：

- 测试ID:一个唯一的代码（如TC_001），便于跟踪。
- 描述:目标，例如使用有效输入进行验证。
- 前提条件：开始之前需要什么，例如出现在搜索页面上。
- 步骤:要执行的操作，例如选择明天的日期，然后点击“搜索”。
- 预期结果:期望的结果，比如按价格排序的航班列表。
- 后条件:系统将显示结果页面。

阅读更多内容:[如何编写测试案例？]

### 步骤 4：设置测试环境

测试环境应与生产环境非常相似。

- 安装所需的应用程序。
- 配置特定于项目的设置。
- 确保测试数据的可用性。
- 检验硬件和软件要求。

### 步骤 5：执行测试案例

逐步执行测试案例，并以用户身份与应用程序交互。

- 实际结果:执行期间发生的情况。
- 通过/失败状态:实际结果是否与预期结果匹配。
- 观察结果:任何意外行为或可用性问题。

## 步骤 6：日志和报告缺陷

当测试失败或出现意外行为时，使用以下内容记录缺陷：

- 缺陷ID：唯一标识符。
- 摘要：简要说明实际缺陷是什么
- 复制步骤:触发问题的详细步骤。
- 实际与预期结果：发生的情况与本应发生的情况。
- 严重性：检查影响是严重、严重还是轻微。
- 附件：用于证明缺陷的屏幕截图、日志或视频。

## 步骤 7：跟踪和检验缺陷

应用修复后：

- 跟踪工具中的缺陷状态。
- 重新测试已修复的问题。
- 根据结果关闭或重新打开缺陷。

## 步骤 8::进行回归测试

回归测试可确保修复缺陷或进行新的更改不会中断现有功能。

- 在解决Bug后会检查受影响的区域。
- 检查关键功能。
- 检查集成点，确保它们像以前一样工作。

## 步骤 9：准备测试结束报告

测试完成后，根据测试计划的目标计算结果，并为相同项目创建测试结束报告：

- 摘要:测试活动概述。
- 测试结果：已执行、通过和失败的测试用例数。
- 发现缺陷:全部缺陷、其严重性和解决状态。
- 未决问题:任何未解决的缺陷或风险。
- 所学课程:对未来测试的见解。

## 步骤 10：提供反馈和建议

分析测试结果，为利益相关者提供切实可行的反馈，例如：

- 软件质量。
- 流程改进。
- 未来的测试策略。

- 用户体验见解。

## 用于手动测试的工具

- TestRail：用户友好的测试管理工具，通过强大的集成和控制面板来组织、执行和报告手动测试案例
- Xray（适用于Jira）：基于Jira的测试管理工具，支持手动、自动化和BDD测试，具有完全的可跟踪性和无缝集成
- Qase：基于云的现代测试管理平台，具有简单的UI、AI支持的测试用例创建和内置的缺陷跟踪
- Zephyr：可扩展的测试管理解决方案，支持具有强大Jira集成和报告功能的手动和探索性测试
- Tuskr：一种轻巧且经济实惠的基于云的测试管理工具，具有直观的界面和协作功能。

## 需要手动测试

- 无缺陷和稳定性：手动测试的主要目标是确保应用程序无缺陷、稳定、符合要求，并向客户提供稳定的产品。
- 熟悉产品：手动测试有助于测试工程师更加熟悉该产品，并获得最终用户视角。这有助于他们为软件编写正确的测试案例。
- 修复缺陷：手动测试有助于确保缺陷已由开发人员修复，并且已针对已修复的缺陷执行了重新测试。

## 手动测试的优势

- 快速准确的可视反馈：它可检测软件应用程序中的几乎所有Bug，并用于测试动态变化的GUI设计，如布局、文本等。
- 成本较低：由于无需任何高水平技能或特定类型的工具，因此成本较低。
- 无需编码：使用黑盒测试方法时无需编程知识。新测试者很容易学习。
- 有效应对计划外的更改：如果应用程序发生计划外的更改，则适合手动测试，因为可以轻松采用手动测试。




卡塔隆

- 功能测试:验证API是否正确执行其预期操作，按照指定处理输入、输出和状态代码。
- 性能测试:评估API在各种负载条件（例如峰值流量、应力）下的速度、稳定性和可扩展性。
- 安全测试:识别漏洞，例如SQL注入、跨站点脚本(XSS)和中断的身份验证/授权，以保护敏感数据。
- 集成测试:确认API交互的系统不同部分或外部服务无缝地协同工作。
- 合同测试:确保API遵守协定的合同（规范，如OpenAPI/Swaggeror WSDL），防止在服务更新之间中断更改。
- 端到端测试:验证涉及链接在一起的多个API调用的整个用户过程。

- 工作原理

可视化无代码工具可让您轻松创建、扩展和组织跨API、Web UI、数据库、ESB甚至常用于AI输入系统中的MCP服务器的测试。不需要深厚的技术技能。SOAtest支持120多种协议和消息格式，为您提供统一的框架来端到端验证业务逻辑。

使用SOAtest，您可以：

- 创建基于方案的流，以模拟实际业务事务，帮助您查找由特定序列触发的隐藏漏洞。
- 以最少的技术专业知识构建测试逻辑、复杂断言、循环和数据驱动流程。
- 运行单个测试或完整套件，并附加回归控制以立即捕获意外更改。

# JavaScript Statements

```
Multiple statements on one line are allowed.
```

# JavaScript Statements

Multiple statements on one line are allowed.

# JavaScript Statements

Multiple statements on one line are allowed.

# JavaScript Statements

Multiple statements on one line are allowed.

# JavaScript Statements

Multiple statements on one line are allowed.

## 什么是软件手动测试？

手动测试是借助软件的各种特性和功能来验证软件的过程。它以预先设计的一组测试为指导，这些测试将对软件进行验证，并提供最终结果报告。此类测试需要一定时间才能完成，因为它完全通过手动操作完成。因此，在执行此类测试时，始终存在一定范围的人为错误。

在采用自动化之前，每个新软件首先都要进行手动测试。手动验证完整的软件会花费更多的时间。一旦软件的所有功能、功能稳定，工作正常，部分人工测试用例就可以转化为自动化。首先评估手动测试用例，检查它们是否能够完全自动化。此类型的测试不需要使用任何自动化工具来完成整个流程。



## 软件手工测试的特点

下面列出了软件手动测试的特征−

- 人工测试完全在人工干预的帮助下进行。
- 探索性测试是手动测试的重要组成部分。在探索性测试中、测试人员无需预先设定任何测试集即可对软件进行验证。检测未预测的缺陷，提高客户满意度。
- 手动测试是灵活的，因为它允许根据需求变化和其他测试条件修改测试案例。
- 手动测试可以从软件开发生命周期(SDLC)的初始阶段开始采用。
- 有些复杂的测试用例只能手动执行，没有任何自动化。
- 手动测试对于验证软件的用户界面非常有用。它有助于验证软件的显示、响应能力和正常设计。

## 为什么需要软件手动测试？

由于下列原因，需要进行软件手动测试−

- 手动测试可确认软件没有任何缺陷，能按要求正确运行，并且足够稳定，可在生产环境中部署。
- 通过手动测试，测试人员可以熟悉该软件，并了解软件如何与客户进行响应。这有助于开发有效的测试案例。
- 手动测试可识别并解决软件中的缺陷。

## 软件手动测试步骤

下面列出了软件手动测试的不同步骤−

第1步−第一步通过阅读需求和规范文档、指南等涉及需求分析阶段。

第2步−第二步包括创建满足所有要求的测试计划。

第3步−第三步包括创建涵盖每项需求的测试案例。

第4步−第4步涉及在正确的测试环境中执行测试案例。

步骤5−第五步涉及分析测试执行结果，并将差异报告为缺陷。

第6步−第6步涉及缺陷修复和重新测试。它还包括重新执行失败的测试用例。

## 软件手动测试的类型

下面列出了不同类型的软件手动测试−

- 黑盒测试−测试人员对软件内部工作一无所知的测试技术。它主要处理验证特性和功能是否按照用户要求正常工作。
- 白盒测试−白盒测试是包括对软件内部结构的验证和软件程序源码的测试过程。
- 灰盒测试−是一种既利用黑盒测试原理，又利用白盒测试技术的测试技术。

## 用于软件手动测试的工具

下面列出了用于软件手动测试的不同工具–

- 测试链接
- 布吉拉
- 吉拉
- LoadRunner
- Apache JMeter
- Perfecto

## 软件手动和自动化测试之间的差异

对软件手工测试和自动化测试进行了比–

| 手动测试 | 自动化测试 |
| --- | --- |
| 它是通过手动操作来验证软件的过程。 | 它是借助自动化工具对软件进行验证的过程。 |
| 它涉及手动执行测试案例。 | 它涉及通过自动化脚本和工具执行测试案例。 |
| 效率较低，需要较多时间才能完成。 | 这种方法效率更高，并且完成所需的时间更少。 |
| 它不能确保百分之百的测试覆盖。 | 与手动测试相比，它可以确保更大的测试覆盖范围。 |
| 它不需要编程技能。只有在了解软件的情况下才能执行。 | 它需要编程技能。 |

## 软件手动测试的优势

下面列出了软件手动测试的优势–

- 手动测试有助于验证屏幕上动态变化的元素。
- 人工测试成本低廉，并且不依赖于熟练的资源。
- 该手动测试可以由没有编程知识的测试器来执行。
- 手动测试可以非常快速地采用，并且适合应对软件中无法预料的更改。

## 软件手动测试的缺点

下面列出了软件手动测试的缺点–

- 手动测试不是很可靠，并且为人为错误提供了范围。
- 需要为不同的模块开发单独的手动测试用例集，使得可重用性范围大大降低。
- 手动测试完全依赖于手动执行测试。但是，有些测试步骤无法通过手动操作执行。

- 执行手动测试的测试人员应具有使用软件的经验。此外，还不能保证在执行手动测试时，软件的所有功能都已被覆盖。
- 手动测试通常是一项耗时的活动。

## 结论

有关软件手动测试教程的全面学习到此结束。首先介绍了什么是软件手工测试，什么是软件手工测试的特点，为什么要进行软件手工测试，软件手工测试的不同步骤是什么，有哪些不同类型的软件手工测试，使用哪些不同的工具进行软件手工测试，软件手工测试与自动化测试有什么区别，软件手工测试的优势是什么，软件手工测试的缺点是什么。这将使您深入了解“软件手动测试”。继续实践您所学到的知识并探索与软件测试相关的其他知识，是加深您的了解并拓展您的视野的明智之举。

## 什么是辅助功能测试？

无障碍环境测试是可用性测试的一个子集，考虑的用户是拥有各种能力和残疾的人。此测试的意义在于检验可用性和可访问性。

可访问性旨在满足不同能力的人的需求，例如：

- 视力障碍
- 物理损伤
- 听力障碍
- 认知障碍
- 学习障碍

一个好的网络应用应该满足所有群体的需求，而不仅仅局限于残疾人。这些新发展包括：

1. 通信基础设施较差的用户
2. 年长者和新用户，他们往往对计算机一无所知
3. 使用旧系统的用户（无法运行最新软件）
4. 使用非标准设备的用户
5. 具有受限访问权限的用户

# 如何执行辅助功能测试

Web辅助功能计划(WAI)描述了对网站进行初步和一致性审核的战略。Web辅助功能计划(WAI)包括一系列软件工具，用于协助进行一致性评估。这些工具包括色盲等具体问题，以及执行自动螺旋成形工具的工具。

# Web辅助功能测试工具

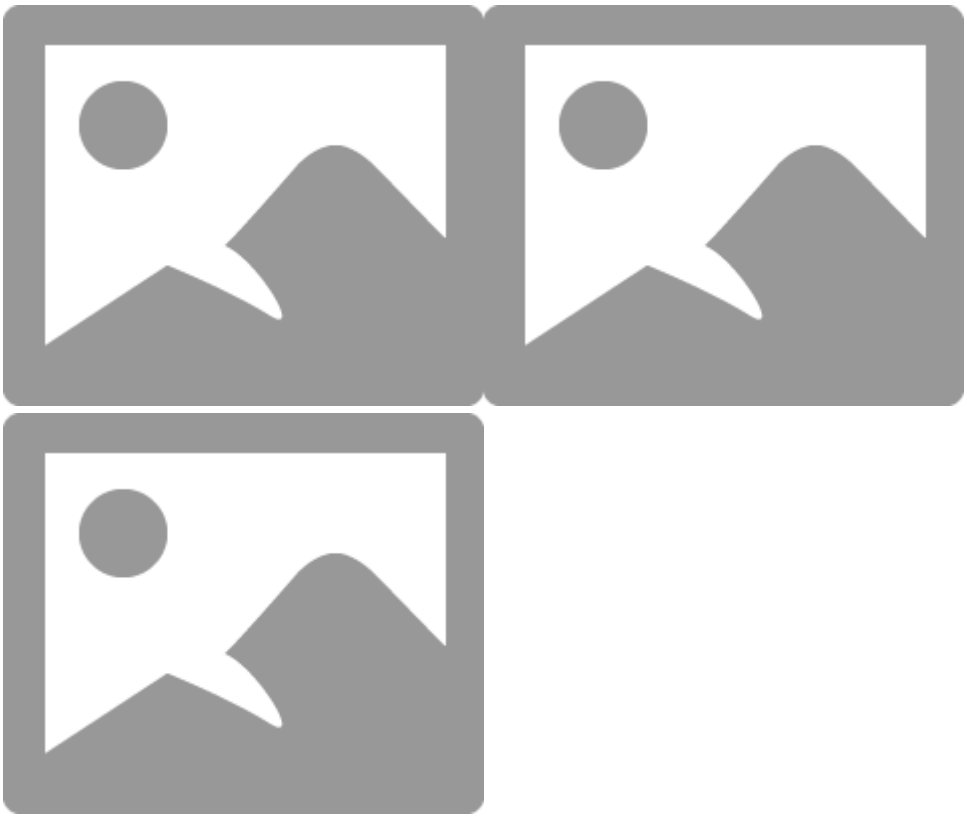
| 产品 | 供应商 | URL |
|---|---|---|
| AccVerify | HiSoftware | http://www.hisoftware.com |
| 鲍比 | Watchfire | http://www.watchfire.com |
| WebXM | Watchfire | http://www.watchfire.com |
| 上升斜坡 | Deque | http://www.deque.com |
| 焦点集中 | SSB技术 | http://www.ssbtechnologies.com/ |

# 自动化工具在验收测试中的作用

上述自动可达性测试工具非常善于识别需要手动检查可达性的代码页和行。

1. 检查站点代码的语法
2. 搜索人类列出的已知模式
3. 标识包含可能导致问题的元素的页面
4. 确定一些实际的可访问性问题
5. 确定一些潜在问题

要解释自动无障碍测试工具的结果，需要具备无障碍技术方面的经验，同时了解技术和可用性问题。



测试以正式和非正式的方式进行，以提高软件质量。正式测试完成后，会进行一轮非正式及任意测试。这称为临时测试。

## 什么是临时测试？

即席测试是在软件上为查找缺陷而执行的非正式测试技术。它是以随机格式进行的，也称为猴测试。临时测试不采用系统化的方法，没有任何有详细记录的测试案例。

临时测试没有任何文档、测试场景、案例等。开发者发现很难修复临时测试中发现的缺陷，因为这些测试文档不存在。此外，某些关键、罕见和意外漏洞只能通过对软件进行随机非正式测试来识别。它也是一种验收测试，可以节省创建新测试用例的时间。

即席测试的一个实例是假定一个软件需要在一天之内发往客户端，并且它的开发在一天之前完成，此时没有时间创建和执行测试用例，因此测试小组根据总体产品知识和经验对整个软件进行即席测试。

# 临时测试的类型

以下列出了不同类型的ad hoc测试−

## 好友测试

在伙伴测试中，在测试过程中至少有两名成员参与 — 一名开发人员和一名测试人员。一旦开发人员完成了组件的实施，他就对其执行单元测试。在测试员将一些随机、任意的数据提供给同一组件后检查结果。如果出现任何错误，开发人员会修正这些缺陷。

## 线对测试

在配对测试中，有两个测试者参与。其中一组对软件进行非正式的、随机的验证，另一组记录测试结果。因此，两者成对工作，交流思想、知识，以便正确完成测试。

# Ad Hoc测试的特点

即席测试的功能如下所列−

- 这是一种随意而非正式的测试方式。
- 任何文档、测试方案、案例等均不支持此功能。
- 它会在正式测试完成后执行。
- 它不遵循系统或结构化的方法。
- 执行临时测试所需的时间更少。
- 当测试用例不可用时，它会检测软件上的错误。

# 临时测试何时完成？

即席测试在以下列出的场景中完成&minu;

- 可用于测试软件的时间有限。
- 正式测试已完成。
- 测试用例不可用。

# 何时未完成临时测试？

在下面列出的场景中不会进行临时测试−

- 如果通过执行测试用例检测到错误，则不会执行此操作。
- 在进行beta测试时，不会执行此测试。

## Ad Hoc测试的优势

即席测试的优势如下−

- 它不遵循任何流程，因此可在软件开发生命周期中的任何时间点执行临时测试。
- 测试团队可以应用新的测试技术来验证软件和寻找错误，而不只是依赖测试用例。
- 开发人员可以在他正在开发的同一模块上执行临时测试，并提高其代码质量。
- 尽管正式的测试过程需要大量的时间，但是临时测试可以在很短的时间内完成。
- 它不需要任何文档。

## Ad Hoc测试的缺点

临时测试的缺点如下所列−

- 临时测试需要由具有测试经验和对产品有充分了解的团队成员来执行。任何经验不足的团队成员都无法执行临时测试。
- 如果存在Bug，则难以复制相同的内容，因为临时测试并非由任何计划来驱动。

## 临时测试中应遵循的最佳实践

下面列出了临时测试中应遵循的最佳实践−

- 收集有关产品的所有知识。
- 确定软件容易出现缺陷的组件并排定其优先级。
- 使用合适的测试工具。

## 结论

有关软件Ad Hoc测试教程的全面介绍到此结束。我们从介绍什么是临时测试开始，什么是临时测试的类型、功能、技术、优点、缺点、时间和最佳实践。

这将使您深入了解软件即席测试。继续实践您所学到的知识并探索与软件测试相关的其他知识，是加深您的了解并拓展您的视野的明智之举。

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。