

河南中医药大学信息技术学院（智能医疗行业学院）智能医学工程专业《互联网医疗服务开发》课程

第10章：Web测试与发布

冯顺磊

河南中医药大学信息技术学院（智能医疗行业学院）
河南中医药大学信息技术学院智能医疗教研室
<https://aitcm.hactcm.edu.cn>
2025/12/3

本章概要

- Web测试
- Web发布



1. Web测试

1.1 Web测试定义

- Web凭借其广泛性、交互性和易用性等优势在全球范围内迅速普及，网页数量呈现出指数级的增长态势。吸引尽可能多的用户并使其长时间关注，是网站追求的核心目标，也是衡量网站成功与否的关键指标。通过Web测试，能够优化Web前端性能，提升其准确性与安全性，因此，Web测试是Web应用开发过程中不可或缺的重要环节。
- 鉴于Web具有分布性、异构性、并发性和平台无关性等特性，Web测试相较于普通程序测试更为复杂，需要从多个维度、多个角度进行全面测试。
- Web测试隶属于软件测试领域，是针对Web服务特性开展的软件测试工作。由于Web应用直接面向用户，通常需要承受长时间的大量操作，因此Web项目的功能和性能必须经过严格验证，这就要求对Web项目进行全面测试。
- Web测试的难点在于Internet和Web媒体的不可预测性。例如，测试人员难以精准界定用户的网络接入状况、所使用的浏览器和操作系统、客户计算机的类型和配置信息、用户所在的国家和地区、使用的语言以及个人文化宗教信仰等，这些因素使得Web测试的难度进一步增加。

1. Web测试

1.2 Web测试内容

- 通常，Web测试可分为以下六个部分。
 - 功能测试
 - 功能测试旨在对Web的具体功能进行全面检测，主要涵盖链接测试、表单测试、数据验证测试、Cookies测试、Web支持系统（如数据库等）测试以及特定功能流程（例如电子商城中的下单、购买、支付、验收、确认等全流程）测试等。
 - 性能测试
 - 性能测试主要针对Web在高并发、高压环境下的服务状况进行评估，主要包括连接速度测试、负载测试、压力测试等。
 - 用户界面测试
 - 用户界面测试着重对Web的UI进行系统性检测，以确保用户访问的UI能够准确、正常地传递Web信息。主要包括导航测试、图形测试、动画测试、内容测试以及用户交互测试等。

1. Web测试

1.1 Web测试定义

- 通常，Web测试可分为以下六个部分。
 - 兼容性测试
 - 兼容性测试主要是考虑到Web访问者的不可预测性而进行的测试，旨在确保任何用户在任何地点通过各种终端设备都能正常访问Web。主要包括操作系统兼容性测试、浏览器兼容性测试、分辨率兼容性测试、以太网接入环境兼容性测试、多智能终端兼容性测试、多语言支持测试等。
 - 安全测试
 - 安全测试主要针对Web安全性和表单安全性进行检测，以保障Web能够稳定、可靠地提供服务。安全测试包括传输安全、表单安全、日志安全、脚本安全、业务接口安全等方面的测试。
 - 接口测试
 - Web通常并非孤立存在，往往会涉及大量对外部服务的调用。例如，位置服务的Web会调用Google、百度等地图服务，电子商务网站会调用信用卡、支付网关等服务。因此，需要对Web对外部数据接口的调用进行全面测试，以保障业务的可用性和安全性。

1. Web测试

1.3 Web测试目的

- 由于Web应用结构复杂且媒介特征不可预测，其故障分布和故障原因较为复杂，要发现、分析和排除故障，通常需要进行多方面的测试。
- Web测试不仅要检查和验证Web应用是否按照设计要求运行，还要测试Web应用在不同终端的显示是否正常、是否可用以及是否安全。
- 一般而言，Web测试的目的主要体现在以下几个方面。
 - 验证Web需求和功能是否完整实现，以及在正常和非正常情况下的功能显示状态；
 - 发现Web的缺陷、错误和不足，进而较为准确地预估Web应用潜在的缺陷数量，获取Web应用的质量信息；
 - 根据当前发现的问题进行分析，为Web应用的下一版本开发提供支持；
 - 发现影响用户使用的错误，预防用户访问或使用Web应用时可能出现的问题；
 - 通过分析测试结果数据、测试问题记录等数据，了解并剖析Web应用存在的问题，提高Web开发效率；
 - 验证Web是否具备发布和使用的条件。

2. 界面测试

2.1 导航测试

- 用户界面测试（User interface testing），简称 UI 测试，旨在检验用户界面功能模块的布局是否合理、整体风格是否统一、各控件的放置位置是否契合客户使用习惯。尤为重要，需满足用户操作便捷、导航简明易懂、界面文字准确无误、命名规范统一、页面美观大方以及图文排版整洁等基本要求。
- 界面作为软件与用户交互的最直接层面，其优劣直接决定了用户对软件的第一印象。界面测试的目标在于确保用户界面能为用户提供恰当的访问和浏览测试对象功能的操作途径。一般而言，界面测试主要涵盖导航测试、图形测试、内容测试和整体界面测试。

2. 界面测试

2.1 导航测试

- 导航体现了用户在页面内的操作方式，包括在不同的用户接口控制（如按钮、对话框、列表和窗口等）之间，以及在不同的连接页面之间的操作。通过考量以下问题，可判断一个 Web 应用系统的导航是否便捷：导航是否直观、Web 系统的主要部分能否通过主页访问、Web 系统是否需要站点地图、搜索引擎或其他导航辅助工具等。
- 在页面上堆砌过多信息往往会适得其反。Web 应用系统的用户多为目标驱动型，他们会快速浏览系统，寻找满足自身需求的信息，若未找到便会迅速离开。极少有用户愿意花费时间去熟悉 Web 应用系统的结构，因此 Web 应用系统的导航应尽可能精准。
- 导航的另一关键要素是 Web 应用系统的页面结构、导航、菜单和连接的风格是否一致。要确保用户能凭直觉判断系统内是否还有其他内容以及内容的位置。
- 一旦确定 Web 应用系统的层次结构，就应着手测试用户导航功能，让最终用户参与测试，效果将更为显著。



2. 界面测试

2.2 图形测试

- 在 Web 应用系统中，恰当的图片 and 动画既能起到广告宣传的作用，又能美化页面。Web 应用系统的图形元素包括图片、动画、边框、颜色、字体、背景和按钮等。图形测试主要包含以下五个方面。
 - 确保图形具有明确的用途，避免图片或动画随意堆砌。Web 应用系统的图片尺寸应尽量小，且能清晰传达信息，通常需链接到具体页面。
 - 验证所有页面字体的风格是否统一。
 - 背景颜色应与字体颜色和前景颜色相协调。
 - 图片的大小和质量至关重要，一般采用 JPG、PNG 格式进行压缩，尽量将图片大小控制在 50kb 以下。
 - 验证文字与图片的混排是否正确。若说明文字指向右侧图片，应确保该图片位于右侧。避免因使用图片导致窗口和段落排列异常或出现孤行。



2. 界面测试

2.3 内容测试

- 内容测试用于检验 Web 应用系统所提供信息的正确性、准确性和相关性。
- 信息的正确性指信息是否可靠，而非误传。例如，商品价格列表中的错误价格可能引发财务问题甚至法律纠纷；信息的准确性指是否存在语法或拼写错误，此类测试通常借助文字处理软件完成，如使用 Microsoft Word 的“拼音与语法检查”功能；信息的相关性指当前页面是否能提供与当前浏览信息相关的信息列表或入口，即一般 Web 站点中的“相关文章列表”或“相关推荐”。

2. 界面测试

2.4 整体界面测试

- 整体界面指整个 Web 应用系统的页面结构设计，它能给用户带来整体的感受。例如，用户浏览 Web 应用系统时是否感到舒适、能否凭直觉找到所需信息、整个 Web 应用系统的设计风格是否一致等。
- 对整体界面的测试过程，实际上是对最终用户进行调查的过程。一般 Web 应用系统会在主页设置调查问卷，以收集最终用户的反馈信息。
- 对于所有的用户界面测试而言，都需要外部人员（与 Web 应用系统开发关联较少或无关联的人员）参与，最好能有最终用户参与。

3. 兼容性测试

3.1 平台兼容性测试

- 基本原理
 - 市场上存在多种不同类型的操作系统，常见的有 Windows、Unix、Mac、Linux 等。Web 应用系统的最终用户使用何种操作系统取决于其系统配置，而 Web 开发者需确保 Web 应用在所有用户眼中都能呈现完美效果，因此需要查看页面在不同平台和不同浏览器下的测试截图。
- 测试目的
 - 平台测试的主要目的是确保被测项目能在不同的操作系统平台（包括同一操作系统的不同版本）上正常运行。

3. 兼容性测试

3.2 浏览器兼容性测试

- 基本原理
 - 浏览器是 Web 客户端的核心组件，不同厂商的浏览器对 HTML 标签（如表格间距、框架处理）、CSS 样式表（如编写规范）、JavaScript（网页元素名称、方法名称等）、ActiveX 控件、浏览器插件（plug - ins）和安全性等诸多方面的支持程度各异。此外，框架和层次结构风格在不同浏览器中的显示效果也有所不同，甚至可能无法显示。
- 测试目的
 - 浏览器兼容性测试主要是在不同的浏览器环境下对 Web 应用的显示效果进行测试，确保 Web 应用的功能能在广泛的浏览器软件上正常运行，保证用户无论使用何种浏览器都能正确访问 Web 应用。

3. 兼容性测试

3.3 分辨率兼容性测试

- 基本原理
 - 由于用户终端设备不同，用户环境的分辨率也存在差异，而分辨率对 Web 页面的展示效果影响显著，不同大小的分辨率会使页面排版、字体样式等的显示产生明显变化。
- 测试目的
 - 对 Web 系统的页面进行分辨率测试，以确定用户使用操作系统的分辨率是否会影响 Web 应用的正常显示，并为页面优化提供依据和建议，最终提升 Web 应用的整体性能，提高用户体验满意度。



4. 功能测试

4.1 链接测试

- 功能测试旨在依据规格说明的要求，确保功能准确无误。其涵盖的内容包括 HTML 语法检查、链接检查、表格测试，以及发送请求和处理服务回传信息等。
- 根据测试要求的难易程度，功能测试可细分为简单功能测试、任务特征测试、边界测试、强制错误情况测试和探测性测试等，以保障不同层次的网站或网络应用程序的运行质量。简单功能测试主要检查链接的可达性；任务特征测试会依据任务的交互性、不确定性等特征进行针对性测试；边界测试会从输入数据域的边界抽取数据开展测试；强制错误情况测试会依据设计规格说明，人为输入明显错误的数据，进而观察系统的运行状况，主要测试系统的容错能力；探测性测试则是边设计边执行测试，试探性地推进并及时调整。
- 相较于其他类型的测试，网站功能测试所需的测试环境要素较少，占用的测试资源也相对较少，只需需求文档明确界定各项功能即可。



4. 功能测试

4.1 链接测试

- 基本原理
 - 链接是 Web 应用系统的核心特征之一，是页面间切换以及引导用户访问其他页面的主要方式。链接测试可从三个方面展开。
 - ① 检查所有链接是否准确链接到指定页面；
 - ② 核实链接目标页面是否存在；
 - ③ 确保 Web 应用系统中不存在孤立页面，即没有任何链接指向的页面。
 - 链接测试必须在集成测试阶段完成，也就是在整个 Web 应用系统的所有页面开发完毕后进行。进行链接测试时需注意以下几点。
 - ① 链接分为站内和站外链接。在测试环境中，站外链接有时无法访问，因此需区分链接失败是由 Bug 导致还是测试环境限制所致。
 - ② 表单链接属于特殊链接，要确认其传输的参数是否正确。
 - ③ 部分链接由客户端程序生成或控制，甚至是动态的。这类链接的测试最好在测试客户端程序时进行。
 - 为保证 Web 应用系统的完整性和可靠性，必须采用穷举法对所有页面的链接情况进行测试。若手工测试工作量大且枯燥，可考虑使用自动化测试工具完成测试工作。
- 测试目的
 - 对网站进行链接测试，可确保页面链接能正常连接到指定文件，提升页面的安全性和加载效率，优化页面搜索性能，提高网站搜索指数。



5. 性能测试

5.1 连接速度测试

- 性能测试借助自动化测试工具，模拟多种正常、峰值及异常负载条件，对系统的各项性能指标展开测试。在软件质量保证体系中，性能测试发挥着关键作用，其涵盖的测试内容丰富多元。中国软件评测中心将性能测试归纳为三个维度：客户端应用性能测试、网络应用性能测试以及服务器端应用性能测试。通常而言，这三个方面实现有效且合理的结合，能够达成对系统性能的全面剖析以及对瓶颈的精准预测。
- 性能测试旨在验证软件系统是否契合用户所提出的性能指标，同时发掘软件系统中潜藏的性能瓶颈，进而对软件进行优化，最终实现系统性能的提升。其具体目的包含以下几个方面。
 - 评估系统能力：测试获取的负荷与响应时间数据，可用于验证既定模型的能力，并为决策提供依据。
 - 识别体系弱点：通过将受控负荷提升至极端水平并突破阈值，从而定位并修复体系中的瓶颈或薄弱环节。
 - 系统调优：反复开展测试，验证系统调整举措是否达成预期效果，进而优化系统性能。
 - 检测软件问题：长时间的测试执行可能引发因内存泄漏导致的程序故障，从而揭示程序中隐匿的问题或冲突。
 - 验证稳定性（resilience）与可靠性（reliability）：在生产负荷条件下持续执行一定时长的测试，以评估系统的稳定性和可靠性是否满足要求。



5. 性能测试

5.1 连接速度测试

- 基本原理
 - 网页包含图片、CSS、JS等多种元素，用户在打开网页时，常遭遇卡顿现象。排除本地网络不稳定因素，多数情况是由网页中的JS和图片等外部因素所致。
 - 用户连接Web应用系统的速度因上网方式而异，可能采用电话拨号或宽带上网。若Web系统响应时间过长，用户很可能因缺乏耐心而放弃访问。
 - 部分页面设有超时限制，若响应速度过慢，用户可能在浏览内容前就需重新登录。此外，连接速度迟缓还可能导致数据丢失，使用户无法获取真实页面。
 - 为精准评估网站在不同网络环境下的访问速度，有必要对网站访问速度进行精确测试，助力开发者分析网站在全球及国内各省份地区的访问状况，找出影响网页加载速度的因素，实现网站的优化。
- 测试目的
 - 通过网站速度测试，掌握不同网络环境下网站的访问速度情况。依据网站速度测试服务提供的网页加载过程分析数据，找出导致网页加载失败或加载缓慢的根本原因，调整网站服务器网络环境、修改并完善程序代码，以提升网站访问速度。

6. Web Vitals

6.1 概述

- 优化用户体验质量是所有网站在网络领域长期取得理想成效的关键所在。无论企业、营销人员还是开发者，网页指标都能助力量化网站的用户体验，并洞察提升的契机。
- “网页指标”计划由 Google 推出，旨在为哪些质量信号对提供卓越的网站用户体验至关重要给出统一的指导方针。
- 多年来，Google 提供了众多用于衡量和报告效果的工具。部分开发者善于运用这些工具，而另一些开发者则因工具和指标繁多，难以跟上其更新节奏。
- 网站所有者无需成为效果专家，也能够了解自身为用户提供的体验质量。Web Vitals 计划致力于简化环境，协助网站聚焦于最重要的指标，即 Core Web Vitals。

6. Web Vitals

6.2 核心网页指标

- 核心网页指标是适用于所有网页的一部分网页指标，所有网站所有者均应衡量这些指标，且这些指标会呈现在所有 Google 工具中。核心网页指标中的每个指标分别代表着用户体验的不同维度，可在实际环境中进行衡量，并反映出以用户为中心的关键结果的真实体验。
- 构成 Core Web Vitals 的指标会随时间推移而演变。当前这组指标着重关注用户体验的三个方面：加载速度、互动性和视觉稳定性。

6. Web Vitals

6.2 核心网页指标

- Largest Contentful Paint (LCP): 用于衡量加载性能。为提供优质的用户体验，LCP 应在网页首次开始加载后的 2.5 秒内完成。
- Interaction to Next Paint (INP): 用于衡量互动性。为提供优质的用户体验，网页的 INP 不应超过 200 毫秒。
- Cumulative Layout Shift (CLS): 用于衡量视觉稳定性。为提供优质的用户体验，网页的 CLS 应维持在 0.1 或更低水平。



6. Web Vitals

6.2 核心网页指标

- LCP
 - LCP (Largest Contentful Paint) ，即最大内容渲染，指的是视窗（viewport）内所有可见元素中尺寸最大的文本块或图像的渲染时间，也就是最大内容在可视区域内变得可见的时间点。所谓最大的内容，例如一篇文章中的一大段文字或产品页面上的一张图片，通常是有助于您理解页面内容的最关键元素。
 - LCP 之所以重要，是因为它与用户的首次加载体验直接相关。快速的 LCP 能让用户感受到页面加载流畅，进而提升用户的满意度和留存率。反之，若 LCP 时间过长，用户可能会感到不耐烦，甚至直接离开页面。
- FID
 - FID (First Input Delay) ，即第一次输入事件延迟，指的是从用户第一次交互事件触发到主线程接收并响应该事件的时间间隔。FID 涉及用户与 Web 页面之间的交互性，用于衡量网站操作的流畅程度。它测量的是从用户首次产生交互行为到浏览器响应该操作的持续时间。这些交互行为可以是单击按钮、点击链接或任何基于 JavaScript 的自定义控件操作。
 - FID 之所以重要，是因为它与用户的交互体验直接相关。较低的 FID 意味着页面能够迅速响应用户输入，从而提升用户的满意度和留存率。反之，若 FID 时间过长，用户可能会觉得页面卡顿或无响应，进而影响用户体验。

6. Web Vitals

6.2 核心网页指标

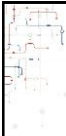
- CLS

- CLS (Cumulative Layout Shift)，即累积布局偏移，指的是首屏页面渲染过程中所有元素结点相对于原始位置的位置偏移累积量。CLS 用于衡量页面布局随时间变化的视觉稳定性，它衡量的是页面生命周期内所有意外布局变化的总和。每次布局变化都会被赋予一个分数，该分数基于变化的大小和影响区域。CLS 即为所有这些分数的累积值。
- CLS 之所以重要，是因为它与用户的视觉体验和交互效率直接相关。较低的 CLS 分数意味着页面布局稳定，用户能够准确地与页面进行交互。反之，若 CLS 分数较高，用户可能会因意外的布局变化而感到困惑或沮丧，甚至引发误操作。

6. Web Vitals

6.3 其他指标

- 尽管 Core Web Vitals 是了解和提供卓越用户体验的关键指标，但仍存在其他辅助指标。
- 这些其他指标可用作三项 Core Web Vitals 的指标代理或补充指标，有助于收集更多的体验数据，或辅助诊断特定问题。
- 例如，首次发送字节时间 (TTFB) 和首次内容渲染 (FCP) 这两个指标均是加载体验的重要方面，对于诊断 LCP 问题（分别为服务器响应时间缓慢或呈现阻塞资源问题）颇为有用。
- 同样，Total Blocking Time (TBT) 等实验室指标对于发现和诊断可能影响 INP 的潜在互动性问题至关重要。不过，由于这些指标无法在现场进行衡量，也无法反映以用户为中心的结果，因此不属于核心 Web 指标集。



6. Web Vitals

6.4 根据Web Vitals优化Web

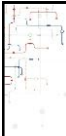
- LCP优化
 - 优化资源加载
 - 确保关键资源（如 CSS、JavaScript、字体和图片）得到优化和压缩，以减小文件大小。
 - 采用合适的图片格式（如 WebP）和图片压缩技术。
 - 对非关键资源（如图片和视频）进行延迟加载。
 - 减少服务器响应时间
 - 优化服务器配置和代码，缩短服务器处理请求的时间。
 - 借助内容分发网络（CDN）来降低网络延迟。
 - 优化渲染阻塞资源
 - 将关键 CSS 进行内联处理，减少渲染阻塞时间。
 - 对非关键 JavaScript 采用异步或延迟加载方式。
 - 优化 DOM 结构
 - 减少 DOM 元素的数量，避免构建复杂的 DOM 结构。
 - 运用现代的前端框架和库，它们通常具备更优的性能优化。
 - 使用高效的缓存策略
 - 合理设置缓存头，利用浏览器缓存减少重复加载时间。



6. Web Vitals

6.4 根据Web Vitals优化Web

- CLS优化
 - 预设尺寸
 - 为图片、视频和广告等可变内容预先设定尺寸（宽度和高度），防止它们在加载时引发布局变化。
 - 避免使用无尺寸元素
 - 避免在页面加载后动态插入无尺寸的元素，这类元素可能会导致布局变化。
 - 使用稳定的字体
 - 在使用网络字体时，确保字体文件能够快速加载，或者采用字体显示策略（如 font-display: swap;）来减少布局变化。
 - 延迟加载非关键资源
 - 对图片、视频和其他非关键资源进行延迟加载，以降低它们加载时对布局的影响。
 - 避免在交互元素附近动态插入内容
 - 避免在按钮、链接等交互元素附近动态插入内容，以减少用户误点击的风险。
 - 使用 CSS Flexbox 和 Grid
 - 采用 CSS Flexbox 和 Grid 布局，它们通常比传统的布局方法更具稳定性。



6. Web Vitals

6.4 根据Web Vitals优化Web

- FID优化
 - 减少 JavaScript 执行时间
 - 优化 JavaScript 代码，减少主线程上的长任务（long tasks），因为这些任务会阻塞事件处理。
 - 利用 Web Workers 处理非 UI 相关的计算，释放主线程。
 - 优化关键渲染路径
 - 确保关键资源（如 CSS 和 JavaScript）得到优化和压缩，减小文件大小。
 - 将关键 CSS 进行内联处理，减少渲染阻塞时间。
 - 对非关键 JavaScript 采用异步或延迟加载方式。
 - 减少 DOM 大小和复杂性
 - 减少 DOM 元素的数量，避免构建复杂的 DOM 结构。
 - 运用现代的前端框架和库，它们通常具备更优的性能优化。
 - 优化资源加载顺序
 - 确保关键资源优先加载，缩短用户交互前的等待时间。
 - 对非关键资源（如图片和视频）进行延迟加载。
 - 使用高效的缓存策略
 - 合理设置缓存头，利用浏览器缓存减少重复加载时间。



7. Selenium

6.4 根据Web Vitals优化Web

- Selenium 作为一款被广泛应用的自动化测试工具，主要用于 Web 应用程序的自动化测试。它支持多种编程语言，涵盖 Java、Python、C#、Ruby、JavaScript 等，能够让用户控制浏览器执行各类自动化任务，包括点击按钮、填写表单以及验证页面内容等操作。此外，Selenium 具备跨平台特性，可在多种浏览器（如 Chrome、Firefox、Edge、Safari 等）以及不同操作系统（如 Windows、macOS、Linux）上稳定运行。
- Selenium 具有以下显著特点：
 - 跨浏览器支持：Selenium 兼容多种主流浏览器，如 Chrome、Firefox、Safari、Internet Explorer 等，可确保相同的测试脚本在不同浏览器环境下均可执行。
 - 多语言支持：Selenium 对多种编程语言提供支持，开发者能够选用自己熟悉的语言编写测试脚本，提升开发效率。
 - 灵活性：Selenium 可与多种测试框架（如 TestNG、JUnit、PyTest 等）无缝结合，为测试工作提供丰富多样的功能。
 - 与 CI/CD 集成：Selenium 能够与 Jenkins 等持续集成工具集成，助力实现自动化测试和持续交付流程。
 - 自动化 Web 交互：Selenium 能够模拟真实用户的操作行为，如点击、滚动、键盘输入等，为测试提供更真实的场景。



7. Selenium

- Selenium 由多个组件构成，每个组件都有其特定的功能和用途。其主要组件如下：
 - Selenium WebDriver：作为 Selenium 的核心组件，用于直接与浏览器进行交互。WebDriver 提供了丰富的 API 接口，开发者可通过代码精确控制浏览器的行为，如打开网页、点击按钮、填写表单等。
 - Selenium IDE：这是一款浏览器插件，主要用于录制和回放用户的操作。对于初学者而言，Selenium IDE 能够帮助他们快速创建简单的测试脚本，但在处理复杂逻辑和条件判断时存在一定局限性。
 - Selenium Grid：这是一个用于并行执行测试的工具。借助 Selenium Grid，可在多个浏览器和操作系统上同时运行测试，有效加快测试速度并提高测试覆盖率。

信创智能医疗系统研发课程体系

河南中医药大学信息技术学院（智能医疗行业学院）



河南中医药大学信息技术学院（智能医疗行业学院）智能医疗教研室

河南中医药大学医疗健康信息工程技术研究所