Chrome Apps是修改过的web apps，人们采用和web平台相同的代码、框架和工具来开发Chrome apps。一些浏览器特性被移除，其它的web API被disabled或者changed用以提高安全性。

一个Chrome App包含了这些组件：

1).manifest:指明app的元信息，告知Chrome你的app，怎么启动等。

2).event page:也称为background script，负责管理app 生命周期.在这里为具体的app事件（如启动和关闭app的窗口）注册监听。

3).code files:必须打包到Chrome App里，包括HTML,CSS,JS和Native Client模块。

4).assets:包括app的图标，也当被打包到Chrome App里。

app container和programming模块 控制Chrome Apps外观和行为， 意在提供给用户一个更本地化的体验。  
Chrome Apps架构：

1.App container model

app container描述Chrome Apps的外观和加载行为。它简单地包含了一个空白的矩形区域这使得app能和系统的本地apps混合，并防止用户通过手动改变URL搞乱app logic。

Chrome Apps加载的内容有：HTML documents(with JS and CSS)； Chrome Apps的加载方式和web apps的不同。两者加载同类型的内容：HTML文档。然而，一个Chrome App会被加载到app container中，而不是浏览器标签页(tab)中，并且app container还必须从本地资源中加载Chrome App 的主文档，这样迫使Chrome Apps在离线时能至少提供最小功能性，也能为实现严格的安全策略提供地方。

2.Programming model

用来描述Chrome Apps 的生命周期和窗口行为。Chrome Apps生命周期应当与浏览器window行为或是一个网络连接相独立。“event page”通过响应用户操作和系统事件来管理Chrome Apps的生命周期，它不可见，只存在于背景中，能由系统runtime自动关闭。一个Chrome App 可以只有一个“event page”。App 生命周期大体包含：Installation、Startup、Termination、Update、Uninstallation。

3.Security model

通过确保用户的信息是通过安全且安心的行为进行管理的方式保护用户。本地化地加载Chrome App主页提供了比从web加载更严格的安全限制。

Chrome App Lifecycle

app runtime和event page 负责管理app的生命周期。app runtime管理app的安装、控制event page并在任意时刻关闭app。event page 监听从app runtime发出的事件，并管理启动什么和怎样启动。

1.lifecycle工作原理

app runtime从用户桌面加载event page，onLaunch()事件触发。这个事件告诉event page启动哪些窗口和窗口的尺寸大小。当event page没有执行JavaScript、没有要处理的回调并且没有开着的窗口时，runtime会unload这个event page并关闭app。在unload之前，会触发onSuspend()事件。这让event page在关闭app之前有机会做些简单的清理任务。

2.创建event page和窗口

所有的app必须有一个event page。这个page在没有自己的UI的情况下包含了应用的顶层逻辑，并负责为其它所有的app pages创建窗口。

1) 创建event page

在app的manifest文件中包含“background”域，并在脚本数组中包含background.js。任何event page使用的库脚本都需要首先被添加到“background”域。event page必须包含onLaunched()函数。这个函数在不管用什么方式启动你的应用时都会被调用。chrome.app.runtime.onLaunched.addListener(function(){});

2) 创建窗口

一个event page根据自己的 判断可能创建一个或多个窗口。Chrome Apps中的windows与Chrome浏览器windows没有关联。

3）including启动数据

默认情况下当app由app launcher启动时是没有启动数据的。

3.监听app runtime事件

1）存储本地设置

event page能监听onInstalled事件，并用Storage API存储和更新本地设置。chrome.runtime.onInstalled.addListener(function(){});

2）防止数据丢失

使用Storage API(storage.sync)，用户能同Chrome sync时自动同步。

3）app关闭前的清理工作

event page能监听onSuspend()事件，并做清理工作和存储app的状态。chrome.runtime.onSuspend.addListener(function(){});

CSP(Content Security policy):

CSP for Chrome Apps限制的操作有：1).在Chrome App页不能使用内联脚本，包括<script>块和事件handlers(<button onclick=”…”>)； 2). app files里不能引用任何外部资源(除video和audio资源外)；3).不能使用string-to-JavaScript方法，如eval()和new Function()。