<img src="/static/images/burger-icon.png" class="collase-ic
on">
Extensions are made of different, but cohesive, components.
Components can include
and various logic files. Extension components are created w
ith web development technologies: HTML, CSS, and JavaScrip
t. An extension's components will depend on its functionalit
y and may not require every option.
This tutorial will build an extension that allows the user
to change the background color of any page on
It will use many core components to give an introductory
demonstration of their relationships.
To start, create a new directory to hold the extension's fi
les.
and include the following code, or download the file
{ "name": "Getting Started Example", "version": "1.
0", "description": "Build an Extension!
The directory holding the manifest file can be added as a
n extension in developer mode in its current state.
Open the Extension Management page by navigating to
The Extension Management page can also be opened by c
licking on the Chrome menu, hovering over
Enable Developer Mode by clicking the toggle switch nex
t to
The extension has been successfully installed. Because no i
cons were included in the manifest, a generic toolbar icon

will be created for the extension.

Although the extension has been installed, it has no inst

ruction. Introduce a

Background scripts, and many other important components,

must be registered in the manifest. Registering a backgrou

nd script in the manifest tells the extension which file t

o reference, and how that file should behave.

{ "name": "Getting Started Example", "version": "1.

0", "description": "Build an Extension!

The extension is now aware that it includes a non-persisten

t background script and will scan the registered file for

important events it needs to listen for.

This extension will need information from a persistent vari

able as soon as its installed. Start by including a listen

ing event for

listener, the extension will set a value using the

API. This will allow multiple extension components to acces

s that value and update it.

chrome.runtime.onInstalled.addListener(function() { chr

ome.storage.sync.set({color: '#3aa757'}, function() {

console.log("The color is green.

field in the manifest for the extension to use them.

{ "name": "Getting Started Example", "version": "1.

0", "description": "Build an Extension!

"background": { "scripts": ["background.js"], "

persistent": false }, "manifest_version": 2 }

Navigate back to the extension management page and click th

e

Click the link to view the background script's console log,

"

This extension uses a button to change the background color

.

<!DOCTYPE html> <html> <head> <style> b

utton { height: 30px; width: 30px;

outline: none; } </style> </head>

<body>

Like the background script, this file needs to be designa

ted as a popup in the manifest under

{ "name": "Getting Started Example", "version": "1.

0", "description": "Build an Extension! ", "permissi

ons": ["storage"], "background": { "scripts": ["ba

ckground.js"], "persistent": false },

Designation for toolbar icons is also included under

, unzip it, and place it in the extension's directory.

Update the manifest so the extension knows how to use the im

ages.

{ "name": "Getting Started Example", "version": "1.

0", "description": "Build an Extension! ", "permissi

ons": ["storage"], "background": { "scripts": ["ba

ckground.js"], "persistent": false }, "page_ac

tion": { "default_popup": "popup.html"

Extensions also display images on the extension management

page, the permissions warning, and favicon. These images

are designated in the manifest under

{ "name": "Getting Started Example", "version": "1.

0", "description": "Build an Extension! ", "permissi

ons": ["storage"], "background": { "scripts": ["ba

ckground.js"], "persistent": false }, "page_ac

tion": { "default_popup": "popup.html", "default

_icon": { "16": "images/get_started16.png",

"32": "images/get_started32.png", "48": "images/get_

started48.png", "128": "images/get_started128.png"

  } },

If the extension is reloaded at this stage, it will inclu

de a grey-scale icon, but will not contain any functionali

ty differences.

is declared in the manifest, it is up to the extension to

tell the browser when the user can interact with

Add declared rules to the background script with the

chrome.runtime.onInstalled.addListener(function() { chr

ome.storage.sync.set({color: '#3aa757'}, function() {

console.log('The color is green.

chrome.declarativeContent.onPageChanged.removeRules(undefin

ed, function() {

conditions: [new chrome.declarativeContent.PageStateMatcher

({

actions: [new chrome.declarativeContent.ShowPageAction()]

{ "name": "Getting Started Example", ... "permiss

ions": [

The browser will now show a full-color page action icon in

the browser toolbar when users navigate to a URL that cont

ains

When the icon is full-color, users can click it to view pop

up.html.

The last step for the popup UI is adding color to the butto

n. Create and add a file called

with the following code to the extension directory, or do

wnloaded

let changeColor = document.getElementById('changeColor');

chrome.storage.sync.get('color', function(data) { chan

geColor.style.backgroundColor = data.color; changeColor.

setAttribute('value', data.color); });

and requests the color value from storage. It then applies

the color as the background of the button. Include a script

tag to

<!DOCTYPE html> <html> ... <body> <button id="changeC

olor"></button>

The extension now knows the popup should be available to us

ers on

and displays a colored button, but needs logic for furthe

r user interaction. Update

let changeColor = document.getElementById('changeColor');

...

chrome.tabs.query({active: true, currentWindow: true}, func

tion(tabs) {

{code: 'document.body.style.backgroundColor = "' + color +

'";'});

The updated code adds an onclick event the button, which

triggers a

This turns the background color of the page the same color

as the button. Using programmatic injection allows for user-

invoked content scripts, instead of auto inserting unwante

d code into web pages.

permission to allow the extension temporary access to the

{ "name": "Getting Started Example", ... "permiss

ions": [

The extension is now fully functional! Reload the extension

, refresh this page, open the popup and click the button t

o turn it green! However, some users may want to change the

background to a different color.

The extension currently only allows users to change the bac

kground to green. Including an options page gives users more

control over the extension's functionality, further cus

tomizing their browsing experience.

<!DOCTYPE html> <html> <head> <style> b

utton { height: 30px; width: 30px;

outline: none; margin: 10px; }

</style> </head> <body>

  <p>Choose a different backgr

ound color!</p>   </body> <script src="op

tions.js"></script> </html>

 to view the options page, although it will currently appe

ar blank.

 Last step is to add the options logic. Create a file called

 in the extension directory with the following code, or do

wnload it

 let page = document.getElementById('buttonDiv'); const kB

uttonColors = ['#3aa757', '#e8453c', '#f9bb2d', '#4688f1'];

  function constructOptions(kButtonColors) { for (let it

em of kButtonColors) { let button = document.createEle

ment('button'); button.style.backgroundColor = item;

  button.addEventListener('click', function() { c

hrome.storage.sync.set({color: item}, function() {

 console.log('color is ' + item); }) });

 page.appendChild(button); } } constructOptions(kBut

tonColors);

 Four color options are provided then generated as buttons o

n the options page with onclick event listeners. When the

user clicks a button, it updates the color value in the ex

tension's global storage. Since all of the extension's files

  pull the color information from global storage no other

 values need to be updated.

 Congratulations! The directory now holds a fully-functional

, albeit simplistic, Chrome extension.

backs up a bit, and fills in a lot of detail about th

e Extensions architecture in general, and some specifi

c concepts developers will want to be familiar with.

Learn about the options available for debugging Extensions

in the

Chrome Extensions have access to powerful APIs above and be

yond what's available on the open web.

has dozens of additional links to pieces of documenta

tion relevant to advanced extension creation.

p class="last-updated">Last updated August 2, 2013.</p