链滴

【UNI-APP 开发 - 必看教程】APP 自动更新（进度条）

作者：luomuren

原文链接：https://ld246.com/article/1654610725854

来源网站：链滴

自动更新：DownLoader

Downloader模块管理网络文件下载任务，用于从服务器下载各种文件，并支持跨域访问操作。通过pls.downloader获取下载管理对象。Downloader下载使用HTTP的GET/POST方式请求下载文件，符标准HTTP/HTTPS传输协议。

方法：

createDownload　新建下载任务

enumerate　枚举下载任务

clear　清除下载任务

startAll　开始所有下载任务

对象：

Download　管理一个下载任务

DownloadEvent　下载任务事件类型

DownloadState　下载任务状态

DownloadOptions　下载任务参数

回调方法：

DownloadCompletedCallback　下载任务完成时的回调

DownloadStateChangedCallback　下载任务状态变化回调

DownloadEnumerateCallback　枚举下载任务回调

使用：plus.downloader.createDownload(url，option，completedCB)

说明：

请求下载管理创建新的下载任务，创建成功则返回Download对象，用于管理下载任务。

参数：

url:（String）必选 要下载文件资源地址

要下载文件的url地址，仅支持网络资源地址，支持http或https协议。 允许创建多个相同url地址的下任务。 注意：如果url地址中包含中文或空格等，需要进行urlencode转换。

options:（DownloadOptions）可选 下载任务的参数

可通过此参数设置下载任务属性，如保存文件路径、下载优先级等。

completedCB:（DownloadCompletedCallback）可选 下载任务完成回调函数

当下载任务下载完成时触发，成功或失败都会触发。

返回值：

新建的下载任务对象

附代码：

```
createDownload(url) {
    var dtask = plus.downloader.createDownload(url, {},
        function(d, status) {
            uni.showToast({
                title: '下载完成',
                mask: false,
                duration: 1000
            });
            // 下载完成
            console.log('status: ' + status);
            if (status == 200) {
                console.log('下载成功：' + d.filename);
                console.log('plus.io.convertLocalFileSystemURL(d.filename): ' + plus.io
                    .convertLocalFileSystemURL(d.filename))
                plus.runtime.install(plus.io.convertLocalFileSystemURL(d.filename), {}, function(succe
s) {
                    uni.showToast({
                        title: '安装成功',
                        mask: false,
                        duration: 1500
                    });
                }, function(error) {
                    uni.showToast({
                        title: '安装失败-01',
                        mask: false,
                        duration: 1500
                    });
                })
            } else {
                uni.showToast({
                    title: '更新失败-02',
                    mask: false,
                    duration: 1500
                });
            }
        });
    try {
        dtask.start(); // 开启下载的任务
        var prg = 0;
        var showLoading = plus.nativeUI.showWaiting("正在下载"); //创建一个showWaiting对象

        dtask.addEventListener('statechanged', function(task, status) {
            console.log('自动更新测试1：',task)
            console.log('自动更新测试2：',status)
            // 给下载任务设置一个监听 并根据状态  做操作
            switch (task.state) {
                case 1:
```

```
                showLoading.setTitle("正在下载");
                break;
            case 2:
                showLoading.setTitle("已连接到服务器");
                break;
            case 3:
                prg = parseInt((parseFloat(task.downloadedSize) / parseFloat(task.totalSize)) *
                    100);
                showLoading.setTitle("  正在下载" + prg + "%  ");
                break;
            case 4:
                plus.nativeUI.closeWaiting();
                //下载完成
                break;
        }
    });
  } catch (err) {
    console.log('错误信息',err)
    plus.nativeUI.closeWaiting();
  }
}
```

思路：

1：初始化，调用后台更新接口

2：获取当前APP应用版本号，和应用版本名称

3：根据接口返回的数据判断是否需要更新，

4：不需要则提示，需要则进行以上代码更新的操作

附官网文档链接：HTML5+ API Reference

版权声明