



در مطلب اول هدف فقط آشنایی و نحوه نصب PouchDB قرار خواهد داشت و در مطالب بعدی نحوه آشنایی با نحوه کدنویسی و استفاده به صورت آفلاین یا آنلاین بررسی خواهد شد .

فهرست مطالب :

بخش اول : معرفی PouchDB

شروع به کار با PouchDB

نحوه استفاده از API ها

سوالات متداول در مورد PouchDB

خطاهای احتمالی

پروژه ها و پلاگین های PouchDB

PouchDB یک دیتابیس NoSQL می باشد که به وسیله Javascript نوشته شده و هدف آن این است که برنامه نویسی ها بتوانند برنامه هایی را توسعه و ارائه کنند که بتواند هم به صورت آفلاین و هم آنلاین سرویس دهی داشته باشند. برنامه اطلاعات خودش را به صورت آفلاین ذخیره می کند و کاربر می تواند زمانیکه به اینترنت متصل نیست، از آنها استفاده کند. اما به محض اتصال به اینترنت، دیتابیس خودش را با دیتابیس آنلاین همگام (Sync) می کند. اینجاست که قدرت اصلی PouchDB مشخص می شود. بزرگترین برتری PouchDB همین است. دیتابیسی است که به صورت توکار قابلیت های همگام سازی را دارا می باشد و به صورت اتوماتیک این کار را انجام می دهد. PouchDB یک پروژه ی اوپن سورس است که توسط [این افراد](#) به روز می شود. البته باید گفت که PouchDB از CouchDB الهام گرفته شده است. اگر شما هم قصد همکاری در این پروژه را دارید بهتر است که [راهنمای همکاری](#) را مطالعه کنید .

پشتیبانی مرورگرها

PouchDB پیش زمینه های مختلفی دارد که به آن این امکان را می دهد تا روی همه مرورگرها و صد البته روی NodeJs کار کند. از IndexedDB بر روی Firefox/Chrome/Opera/IE و WebSql بر روی Safari و همچنین LevelDB بر روی NodeJs استفاده می کند. در حال حاضر PouchDB بر روی مرورگرهای زیر تست شده است:

فایرفاکس 12 و بالاتر

گوگل کروم 19 و بالاتر

اپرا 12 و بالاتر

سافاری 5 و بالاتر

اینترنت اکسپلورر 10 و بالاتر

NodeJs 0.10 و بالاتر

و به صورت شگفت انگیزی در Apache Cordova

برای اطلاعات بیشتر در مورد مرورگرهایی که IndexedDB و WebSql را پشتیبانی می کنند به لینک های زیر مراجعه کنید:

[Can I use IndexedDB](#)

[Can I use Web SQL Database](#)

نکته : در صورتی که برنامه شما نیاز دارد تا از اینترنت اکسپلورر نسخه پایینتر از 10 استفاده کند می توانید از دیتابیسی های آنلاین استفاده کنید، که البته دیگر قابلیت استفاده آفلاین را نخواهد داشت.

وضعیت فعلی PouchDB

PouchDB برای مرورگر، فعلا در وضعیت بتا به سر می برد و به صورت فعالی در حال گذراندن تست هایی می باشد تا باگ های آن برطرف شود و به صورت پایدار (Stable) ارائه گردد. البته فقط ممکن است که شما باگی را در قسمت Api ها پیدا کنید که البته Api ها هم در حال حاضر پایدار هستند و گزارشی مبنی بر باگ در آن ها موجود نیست. اگر هم باگی پیدا بشود شما می توانید PouchDB را بدون ریسک از دست رفتن اطلاعات آپگرید کنید.

PouchDB برای NodeJS فعلا در وضعیت آلفا است و آپگرید کردن ممکن است به اطلاعات شما آسیب بزند. البته با آپدیت به صورت دستی خطری شما را تهدید نخواهد کرد .

نحوه‌ی نصب PouchDB

PouchDB به صورت یک کتابخانه‌ی کوچک و جمع و جور طراحی شده است تا بتواند همه نیازها را برطرف و روی همه نوع Device اعم از موبایل، تبلت، مرورگر و کلا هر چیزی که جاوا اسکریپت را ساپورت می‌کند کار خود را به خوبی انجام بدهد.

برای استفاده از PouchDB میبایست [این فایل را با حجم فوق العاده 97 کیلوبایت دانلود کنید](#) و آن را به یک صفحه وب اضافه کنید :

```
<script src="pouchdb-2.1.0.min.js"></script>
```

آخرین نسخه و بهترین نسخه : [pouchdb-2.1.0.min.js](#)

برای اطلاع از آخرین آپدیتها و نسخه‌ها به [این صفحه در گیت هاب](#) مراجعه کنید .
برای کسانی هم که از NodeJS استفاده می‌کنند نحوه نصب به این صورت است :

```
$ npm install pouchdb
```

نظرات خوانندگان

نویسنده: سعیدزمان
تاریخ: ۱۱:۲۰ ۱۳۹۳/۰۱/۲۶

سلام مهندس مطلب بسیار جالبی بود فقط من یک سوال برام پیش اومده که این اطلاعات که میخواد در حالت افلاین استفاده بشه کجا قرار میگیره ؟ ایا امنیت داده های حساس رو پایین نمی یاره ؟
با تشکر

نویسنده: محمد رضا صفری
تاریخ: ۱۵:۱۵ ۱۳۹۳/۰۱/۲۶

پیشنهاد می کنم این اسلاید هارو ببینید : <http://www.slideshare.net/wurbanski/nosql-no-security>
<http://www.slideshare.net/gavinholt/nosql-no-security-16514872>
واقعا مفیده و خیلی از سوالات شمارو پاسخ میده .
موفق باشید ./

در این آموزش هدف ما ایجاد برنامه‌ای بر اساس [TodoMVC](#) است که می‌تواند خودش را با یک دیتابیس آنلاین همگام سازی کند.

مطمئن باشید بیشتر از 10 دقیقه وقت شمارا نخواهد گرفت !

نصب PouchDB

فایل index.html را باز کنید و فایل‌های PouchDB را به آن اضافه کنید :

```
<script src="//cdn.jsdelivr.net/pouchdb/2.2.0/pouchdb.min.js"></script>
<script src="js/base.js"></script>
<script src="js/app.js"></script>
```

حالا PouchDB نصب شده و آماده به کار است . (در نسخه نهایی بهتر است از فایل‌های local استفاده کنید)

ایجاد بانک اطلاعاتی

بقیه کارها باید در فایل app.js انجام شود . برای شروع باید بانک اطلاعاتی جدیدی را برای درج اطلاعات خودمان ایجاد کنیم . برای ایجاد بانک اطلاعاتی خیلی ساده یک instance جدید از آبجکت PouchDB می‌سازیم .

```
var db = new PouchDB('todos');
var remoteCouch = false;
```

نیازی نیست که برای بانک خود، نما (Schema) ایجاد کنید! بعد از اینکه اسم را مشخص کنید، بانک آماده به کار است.

ثبت اطلاعات در بانک اطلاعاتی

اولین کاری که باید انجام دهیم این است که یک متد را ایجاد کنیم و توسط آن اطلاعات خودمان را در بانک اطلاعاتی ذخیره کنیم. نام متد را addTodo انتخاب می‌کنیم و کارش این است که وقتی کاربر کلید Enter را فشار داد، اطلاعات را داخل بانک اطلاعاتی ذخیره کند. متد مورد نظر به این صورت هست:

```
function addTodo(text) {
  var todo = {
    _id: new Date().toISOString(),
    title: text,
    completed: false
  };
  db.put(todo, function callback(err, result) {
    if (!err) {
      console.log('Successfully posted a todo!');
    }
  });
}
```

در PouchDB هر پرونده نیاز دارد تا یک فیلد unique با نام _id داشته باشد. اگر داده‌ای بخواهد در بانک اطلاعاتی ثبت شود و دارای _id مشابه باشد، با آن به صورت یک update رفتار خواهد شد. در اینجا ما از تاریخ با عنوان id استفاده کردیم که در این مورد خاص می‌باشد. شما می‌تواند از db.post() نیز استفاده کنید؛ اگر یک id را به صورت random می‌خواهید. تنها چیزی که اجباری است در هنگام ساختن یک پرونده، همین _id است و بقیه موارد کاملاً اختیاری هستند.

نمایش اطلاعات

در اینجا ما از یک function کمکی به نام redrawTodosUI استفاده خواهیم کرد که وظیفه‌اش این است تا یک array را دریافت

کرده و آن را هر طور که مشخص کردید نمایش دهد. البته آن را به سلیقه خودتان می‌توانید آماده کنید. تنها کاری که باید انجام دهیم این است که اطلاعات را از بانک اطلاعاتی استخراج کرده و به function مورد نظر پاس دهیم. در اینجا می‌توانیم به سادگی از db.allDocs برای خواندن اطلاعات از بانک اطلاعاتی استفاده کنیم. خاصیت include_docs به PouchDB این دستور را می‌دهد که ما درخواست دریافت همه اطلاعات داخل پرونده‌ها را داریم و descending هم نحوه مرتب سازی را که بر اساس id_ هست، مشخص می‌کند تا بتوانیم جدیدترین اطلاعات را اول دریافت کنیم.

```
function showTodos() {
  db.allDocs({include_docs: true, descending: true}, function(err, doc) {
    redrawTodosUI(doc.rows);
  });
}
```

به روزرسانی خودکار

هر بار که ما اطلاعات جدیدی را در بانک اطلاعاتی وارد می‌کنیم، نیازمند این هستیم تا اطلاعات جدید را به صورت خودکار دریافت و نمایش دهیم. برای این منظور می‌توانیم به روش زیر عمل کنیم :

```
var remoteCouch = false;

db.info(function(err, info) {
  db.changes({
    since: info.update_seq,
    live: true
  }).on('change', showTodos);
});
```

حالا هر بار که اطلاعات جدیدی در بانک اطلاعات ثبت شود، به صورت خودکار نمایش داده خواهد شد. خاصیت live مشخص می‌کند که این کار می‌تواند بی نهایت بار انجام شود.

ویرایش اطلاعات

وقتی که کاربر یک آیتم Todo را کامل می‌کند، یک چک باکس را علامت می‌زند و اعلام می‌کند که این کار انجام شده است.

```
function checkboxChanged(todo, event) {
  todo.completed = event.target.checked;
  db.put(todo);
}
```

این بخش شبیه به قسمت ثبت اطلاعات است، با این تفاوت که باید شامل یک فیلد rev_ (اضافه بر id_) نیز باشد. در غیر اینصورت تغییرات ثبت نخواهد شد. این کار برای این است که اشتباهی، اطلاعاتی در بانک ثبت نشود.

حذف اطلاعات

برای حذف اطلاعات باید از متد db.remove به همراه آبجکت مورد نظر استفاده کنید .

```
function deleteButtonPressed(todo) {
  db.remove(todo);
}
```

مانند بخش ویرایش نیز باید در اینجا هم id_ و هم rev_ مشخص شود. باید توجه داشته باشید در اینجا همان آبجکتی را که از بانک فراخوانی کرده‌ایم، به این متد پاس می‌دهیم. البته شما می‌توانید آبجکت خودتان را نیز ایجاد کنید {id: todo._id, _rev: todo._rev_} اما خوب همان روش قبلی عاقلانه‌تر است و احتمال خطای کمتری دارد .

نصب CouchDB

حالا می‌خواهیم همگام سازی را اجرا کنیم و برای این کار نیاز هست یا [CouchDB را به صورت Local نصب کنیم](#) یا از سرویس‌های آنلاین مثل [IrisCouch](#) استفاده کنید .

فعال سازی CORS

برای اینکه به صورت مستقیم با CouchDB در ارتباط باشید باید مطمئن شوید که CORS فعال هست. در اینجا فقط نام کاربری و رمز ورود را مشخص کنید. به صورت پیش فرض CouchDB به صورت Admin Party نصب می‌شود و نیازی به User و password ندارد. مگر اینکه برایش مشخص کنید. همچنین شما باید myname.iriscouch.com را با سرور خودتان جایگزین کنید که اگر به صورت local کار می‌کنید 127.0.0.1:5984 است.

```
$ export HOST=http://username:password@myname.iriscouch.com
$ curl -X PUT $HOST/_config/httpd/enable_cors -d '"true"'
$ curl -X PUT $HOST/_config/cors/origins -d '"*"'
$ curl -X PUT $HOST/_config/cors/credentials -d '"true"'
$ curl -X PUT $HOST/_config/cors/methods -d '"GET, PUT, POST, HEAD, DELETE"'
$ curl -X PUT $HOST/_config/cors/headers -d \
  '"accept, authorization, content-type, origin"'
```

راه اندازی ارتباط ساده دو طرفه

به فایل app.js برگردید . در اینجا باید آدرس بانک اطلاعاتی آنلاین را مشخص کنیم.

```
var db = new PouchDB('todos');
var remoteCouch = 'http://user:pass@mname.iriscouch.com/todos';
```

فراموش نکنید که نام کاربری و رمز ورود را بسته به نیاز خود تغییر دهید. حالا می‌تونیم function که وظیفه همگام سازی اطلاعات را به عهده دارد بنویسیم :

```
function sync() {
  syncDom.setAttribute('data-sync-state', 'syncing');
  var opts = {live: true};
  db.replicate.to(remoteCouch, opts, syncError);
  db.replicate.from(remoteCouch, opts, syncError);
}
```

db.replicate() به PouchDB می‌گوید که همه اطلاعات را "به" یا "از" remoteCouch انتقال دهد. ما دوبار این درخواست را دادیم. در بار اول اطلاعات به داخل سرور ریموت منتقل می‌شود و در بار دوم اطلاعات local جایگزین می‌شوند.

یک callback هم وقتی که این کار به پایان برسد اجرا خواهد شد . می‌توانید برنامه خود را در دو مرورگر مختلف اجرا کنید تا نتیجه کار را ببینید.

تبریک می‌گوییم !

شما توانستید اولین برنامه خود را توسط PouchDB ایجاد کنید. البته این یک برنامه ساده بود و در دنیای واقعی نیاز هست تا خیلی کارهای بیشتری را انجام دهید. اما باز هم اصول اولیه کار را یاد گرفتید و ادامه راه را می‌توانید تنهایی ادامه دهید .