جافاسكربت - الوحدة العاشرة: أساسيات Node.js و npm

بقيت جافاسكربت حتى الآن داخل المتصفح. أما Node.js فبرنامج مجاني يشغل جافاسكربت على حاسوب عادي أو على خادم، من دون صفحة ويب أصلاً. وتنقلنا أربع خطوات من المتصفح إلى الأتمتة الحقيقية: التشغيل، ثم التثبيت، ثم الخدمة، ثم الأتمتة. وتجيب كل خطوة عن سؤال بسيط. وفي هذه الوحدة يستخدمها سامر، منسق تقنية المعلومات في شركة نقل وخدمات لوجستية، ليعرف ما الذي يملأ القرص المشترك الذي أوشك على الامتلاء.

1. التشغيل

يشغل Node.js ملف جافاسكربت من الطرفية: نكتب الأمر node ثم اسم الملف، فتظهر النتيجة في الطرفية. والسؤال هو: ماذا يحتاج نصي البرمجي من الحاسوب؟ لا توجد صفحة، فلا يوجد document ولا window، لكن توجد وحدات مدمجة للحاسوب نفسه. فوحدة fs تقرأ الملفات وتكتبها، ووحدة path تتعامل مع أسماء الملفات وامتداداتها، ووحدة http تتحدث إلى الشبكة. ونستورد الوحدة بعبارة import مع البادئة node والنقطتين، وينتهي اسم الملف بالامتداد mjs. وكل ما نكتبه بعد اسم الملف يصل في قائمة process.argv، فيعمل النص البرمجي الواحد على أي مجلد نسميه. والكود في الطرفية يملك قوة حقيقية، فهو يستطيع تغيير ملفات حقيقية وحذفها، ولذلك نجرب على نسخة أولاً.

مثال: تنجح وصفة المخبوزات في أي مطبخ، ولا تحتاج إلى واجهة المحل. يتبع الخباز الخطوات المكتوبة على بطاقة، فيخرج الخبز نفسه في كل مرة. والنص البرمجي في Node.js وصفة بلا واجهة محل: يحمل الملف الخطوات، وتكون الطرفية هي المطبخ. نشغله، فتظهر النتيجة هناك مباشرة.

في العمل: يكتب سامر ملفاً اسمه report.mjs. يستورد وحدتي fs و path، ويأخذ اسم المجلد من process.argv، ويطبع سطراً لكل ملف يجده. ويشغله على مجلد تجريبي صغير، لا على القرص المشترك، فتظهر القائمة في الطرفية خلال لحظة.

2. التثبيت

يأتي Node.js ومعه npm، مدير الحزم. والحزمة كود قابل لإعادة الاستخدام كتبه شخص آخر ونشره في سجل npm. والسؤال هو: هل أكتب هذا بنفسي، أم أستخدم حزمة موجودة؟ يبدأ المشروع بملف package.json يسجل اسمه ونصوصه واعتمادياته. وعندما نضيف حزمة، تنزلها npm في مجلد node_modules، وتضيف اسمها إلى الاعتماديات في package.json، وتسجل الإصدارات الدقيقة في ملف package-lock.json. ولا نشارك مجلد node_modules، فالزميل الذي يملك المشروع يشغل الأمر npm install ويعيد بناءه. ونحفظ الأدوات التي لا نحتاج إليها إلا أثناء التطوير باعتبارها اعتماديات تطوير. وكل حزمة كود كتبه غرباء ويعمل بصلاحياتنا، لذلك نفضل الحزم المعروفة والمصونة التي لها اعتماديات قليلة، ونشغل الأمر npm audit الذي يفحص اعتمادياتنا مقابل قائمة بالثغرات المعروفة.

مثال: يشتري البناء الذي يحتاج إلى نافذة نافذة جاهزة من مورد بدل أن يصنع الزجاج بنفسه. لكنه يتحقق من المورد أولاً، لأن النافذة الرديئة تبقى داخل البيت إلى الأبد. والحزمة نافذة جاهزة للكود: توفر علينا العمل، لكنها تصبح جزءاً من مشروعنا، فنتحقق ممن كتبها.

في العمل: يريد سامر عرض الأحجام بالميغابايت والغيغابايت لا بأعداد طويلة من البايتات. ويقرر أن دالة صغيرة من كتابته أسلم من اعتمادية جديدة، فيكتبها. وفي المشروع التالي يضيف أداة اختبار معروفة باعتبارها اعتمادية تطوير. وينسخ زميل المشروع من دون node_modules، ويشغل npm install، فيحصل على الإصدارات نفسها تماماً من package-lock.json.

3. الخدمة

الخادم برنامج ينتظر الطلبات ويرسل الأجوبة. وتنشئ وحدة http في Node.js خادماً في أسطر قليلة. نعطيه دالة تستقبل الطلب والرد، وتضبط رمز حالة، مثل مئتين للنجاح وأربعمئة وأربعة لصفحة غير موجودة، وتنهي الرد بالنص المراد

حقوق الطبع والنشر محمية © - أكاديمية أساسيات (شركة أساسيات للخدمات الاستشارية)

إرساله. ثم يستمع الخادم إلى منفذ، مثل ثلاثة آلاف. والسؤال هو: من يحتاج إلى أن يسأل هذا البرنامج، وماذا ينبغي أن يجيب؟ وما دام الخادم يعمل، نفتح العنوان localhost متبوعاً بالمنفذ في متصفح على الحاسوب نفسه. ولا تصل إليه الحواسيب الأخرى إلا إذا سمحت الشبكة وجدار الحماية، لذلك نتفق على أي خادم للزملاء مع قسم التقنية أولاً. والخادم الصغير يعلمنا كيف يعمل الويب، لكن الخدمة العامة الحقيقية تحتاج إلى أكثر من ذلك بكثير: الأمان والاتصالات المشفرة والمراقبة.

مثال: يجيب مكتب الاستقبال عن نوع واحد من الأسئلة: أين قاعة الاجتماعات؟ يسأل الزائر، ويجيب المكتب، ثم يستطيع الزائر التالي أن يسأل. ولا يبدأ المكتب الحديث أبداً، بل ينتظر السؤال ثم يجيب. وخادم http مكتب استقبال كهذا: ينتظر الطلب، ويرسل جواباً واحداً، ويبقى جاهزاً للتالي.

في العمل: يريد قائد فريق سامر أن يطلع على حالة القرص من دون تشغيل نص برمجي. فينشئ سامر خادماً يبني التقرير عند كل طلب ويرسله صفحة قصيرة. ويفتح العنوان localhost على المنفذ رقم ثلاثة آلاف في حاسوبه، فتظهر الأرقام في المتصفح. ثم يوقف الخادم بالضغط على Control و C، ويتفق مع فريق الشبكة قبل أن يتصل به أي شخص آخر.

4. الأتمتة

النص البرمجي الذي نشغله بأيدينا مفيد، لكن الأمر المسمى أفضل. ففي package.json يربط قسم scripts اسماً قصيراً بأمر. فمثلاً يستطيع نص اسمه report أن يشغل الأمر node report.mjs، ثم يبدؤه الجميع بالأمر `npm run report` من دون أن يتذكروا أسماء الملفات. والسؤال هو: كيف يستطيع أي شخص أن يكرر هذه المهمة بالدقة نفسها غداً أو بعد شهر؟ والتقارير التي تعمل تلقائياً، كل أسبوع مثلاً، هي الخطوة التالية المعتادة، وتتولى الجدولة أدوات نظام التشغيل أو أدوات الشركة.

مثال: تطبع شركة ورقة المبيعات الأسبوعية نفسها. وبدل أن يشرح المدير كيف نبنيها كل مرة، يعطيها اسماً على الرف ويقول: خذوا الورقة الأسبوعية. ولا يسأل الموظف الجديد كيف نصنعها، بل يطلبها باسمها فيحصل على الورقة نفسها. ومدخل النص في package.json هو ذلك الاسم على الرف: كلمة قصيرة واحدة تبدأ المهمة كلها.

في العمل: ينهي سامر التقرير. يقرأ مجلد القرص المشترك ومجلداته الفرعية بوحدة fs، ويسأل عن حجم كل ملف، ويستخدم وحدة path لمعرفة الامتداد، ويجمع المجاميع لكل امتداد في كائن. ويطبعها في جدول بالدالة console.table، مرتبة من الأكبر. ويضيف نصاً اسمه report إلى package.json، فيستطيع الفريق تشغيل npm run report. ويبين الجدول أن فيديوهات تدريب مصدرة من حملة منتهية تملأ نحو ثلثي القرص. فيحفظها الفريق في الأرشيف، ويتأجل شراء مساحة جديدة.

الخاتمة

لم يكن القرص المشترك لغزاً في النهاية. فقد أخرج التشغيل جافاسكربت من المتصفح، وأتاحت لها الوحدات المدمجة قراءة الملفات. وبين التثبيت كيف تجلب npm كود الآخرين عبر package.json، ولماذا نختار بعناية ونشغل npm audit. وحولت الخدمة النص البرمجي إلى برنامج صغير يجيب عن الطلبات. وأعطت الأتمتة التقرير اسماً، فيستطيع أي شخص تكراره. ونبقي كل نص برمجي صغيراً، ونجرب على النسخ، ونقرر قبل كل تثبيت هل نحتاج فعلاً إلى الحزمة.

المصادر

- وثائق Node.js: وحدة نظام الملفات fs (readdir و stat وحجم الملف) – https://nodejs.org/api/fs.html
- وثائق Node.js: وحدة http (createServer و writeHead و end و listen) – https://nodejs.org/api/http.html
- وثائق Node.js: العملية process (process.argv) – https://nodejs.org/api/process.html
- وثائق Node.js: وحدات ECMAScript (الامتداد mjs وحقل type في package.json) – https://nodejs.org/api/esm.html
- وثائق npm: ملف package.json (الاعتماديات واعتماديات التطوير والنصوص) – https://docs.npmjs.com/cli/v10/configuring-npm/package-json
- وثائق npm: الأمر npm audit – https://docs.npmjs.com/cli/v10/commands/npm-audit
- Node.js و npm أسماء تعود إلى أصحابها، وهذا الفيديو مادة تدريبية مستقلة.

حقوق الطبع والنشر محمية © - أكاديمية أساسيات (شركة أساسيات للخدمات الاستشارية)