جافاسكربت - الوحدة الأولى: الخطوات الأولى: المتغيرات وأنواع البيانات

جافاسكربت هي لغة البرمجة التي تشغلها متصفحات الويب، وبها تستجيب الصفحة لما يفعله الزائر. ويبدأ كل برنامج بالحاجة نفسها، وهي حفظ قيمة تحت اسم ليمكن استخدامها مرة أخرى. وتجعل أربع خطوات هذا الأمر آمنا وواضحا، وهي: التصريح، والتسمية، والنوع، والفحص. وتجيب كل خطوة عن سؤال بسيط، وتكفي للتدرب عليها نافذة console المجانية التي يضمها كل متصفح. وفي هذه الوحدة تستخدمها رنا، مساعدة في فريق الطلبات عبر الإنترنت لدى سلسلة مخابز، لتصلح عداد القطع في سلة أحد العملاء.

1. التصريح

المتغير مكان مسمى يحمل قيمة. ننشئه بكلمة محجوزة، ويسمى ذلك التصريح به. والسؤال هو: هل ستتغير هذه القيمة لاحقا؟ إذا كان الجواب لا فنستخدم الكلمة const، وإذا كانت ستتغير فنستخدم الكلمة let. فإذا حاول الكود إعطاء قيمة جديدة لمتغير من نوع const تتوقف جافاسكربت بخطأ، فيظهر الخطأ فورا بدلا من أن يختبئ في نتيجة خاطئة. ومن العادات الجيدة أن نبدأ بالكلمة const ولا ننتقل إلى let إلا عندما يجب أن تتغير القيمة.

مثال: يطبع متجر ساعات العمل على بابه، لكنه يعيد كتابة قائمة المخزون على لوحة كل مساء. نطرح سؤال التصريح عن كل واحد منهما. هل ستتغير ساعات العمل لاحقا؟ لا، ولذلك تشبه const، ولا يعيد أحد طلاء الباب كل يوم. وهل ستتغير قائمة المخزون لاحقا؟ نعم، فهي تتغير باستمرار، ولذلك تشبه let. ويدلنا طرح السؤال عن كل قيمة على الكلمة التي نكتبها.

في العمل: تطرح رنا سؤال التصريح عن كل قيمة في السلة. رسوم التوصيل واحدة في كل طلب فتصرح بها بالكلمة const، أما عدد الكعكات فيزيد مع كل نقرة فتصرح به بالكلمة let. وفي وقت لاحق يحاول تعديل في مكان آخر من الكود أن يكتب فوق الرسوم، فتتوقف الصفحة بخطأ يسمي المشكلة. ولا تصل النتيجة الخاطئة إلى أي عميل، ويرى الزملاء الذين يراجعون الكود في لحظة أي القيم ثابت وأيها قابل للتغير.

2. التسمية

يحتاج كل متغير إلى اسم، ويجب أن يتبع الاسم بعض القواعد. يمكن أن يحتوي على حروف وأرقام والشرطة السفلية وعلامة الدولار، لكنه لا يبدأ برقم ولا يحتوي على مسافات. ولا يمكن استخدام كلمات لها معنى مسبق في جافاسكربت، مثل let و const، أسماء للمتغيرات. والأسماء حساسة لحالة الحروف، فالاسمان deliveryFee و deliveryfee متغيران مختلفان. والأسلوب الشائع يسمى camelCase، فالكلمة الأولى بحروف صغيرة، وتبدأ كل كلمة لاحقة بحرف كبير. والسؤال هو: هل يفهم زميل هذا الاسم دون أن يسألنا؟ فاسم مثل itemCount يشرح نفسه، أما اسم مثل x فيجبر الجميع على التخمين.

مثال: في الجدول الإلكتروني يفهم الجميع عمودا عنوانه رسوم التوصيل، ولا يفهم أحد عمودا عنوانه العمود باء. فيضطر القارئ إلى فتح الخلايا والتخمين. وتعمل المتغيرات بالطريقة نفسها: فمع اسم مثل f يضطر الزميل إلى قراءة البرنامج كله ليعرف ما يحمله. أما مع اسم مثل deliveryFee فالجواب موجود أمام عينيه. فالاسم الواضح توثيق يرافق القيمة حيث ذهبت.

في العمل: كتبت رنا إجمالي السلة في البداية بالاسم t ورسوم التوصيل بالاسم f. وعندما بدت النتيجة خاطئة لم يعرف زميل أي متغير يحمل ماذا. فتطبق سؤال التسمية وتغير الاسمين إلى totalPrice و deliveryFee. وتقول رسالة الخطأ التالية إن deliveryfee غير معرف، فيكتشف الزميل الحرف الكبير الناقص خلال دقائق.

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

3. النوع

لكل قيمة في جافاسكربت نوع، والنوع هو الذي يحدد ما يستطيع الكود فعله بها. السلسلة النصية string هي نص داخل علامات اقتباس، والرقم number يشمل الأعداد الصحيحة والعشرية معا. والقيمة المنطقية boolean إما true وإما false. وتعني undefined أن المتغير موجود وليس له قيمة بعد، أما null فقيمة نضعها عمدا لنقول إنه لا شيء هنا. والسؤال هو: أي نوع من القيم هذه؟ فعلامة الجمع تجمع رقمين لكنها تصل بين نصين، فالرمز نفسه يتصرف بطريقة مختلفة بحسب النوع. وتخزن الأعداد العشرية تقريبيا فقد تظهر فروق دقيقة في التقريب، ولذلك تحسب متاجر كثيرة المبالغ بأصغر وحدة نقدية، مثل السنت، وتحولها في النهاية فقط. والنص القادم من حقل نموذج يكون دائما string، حتى عندما يكتب الزائر رقما.

مثال: رقم الهاتف الذي يبدأ بصفر يبدو رقما، لكن أحدا لا يجمع أرقام الهواتف معا. والنوع الرقمي يسقط الصفر الأول، ولذلك يحفظ رقم الهاتف نصا. أما كمية الكعكات فتدخل في الحساب، ولذلك ينبغي أن تكون رقما.

في العمل: تقرأ رنا الكمية من نموذج الطلب فتصل نصا. وتطرح سؤال النوع، فيفسر الجواب النتيجة الغريبة في صفحة الاختبار. فالسلة تضم كعكتين أصلا، ويكتب العميل ثلاثا أخرى، فتصل علامة الجمع بينهما جنبا إلى جنب، ويظهر العداد ثلاثة وعشرين بدلا من خمسة. وعداد كهذا يربك أي عميل.

4. الفحص

بدلا من التخمين نسأل عن النوع. يعيد العامل typeof نصا يسمي نوع القيمة، مثل string أو number أو boolean. ونطبع النتائج بالأمر console.log ونقرأها في نافذة console بالمتصفح. ولتحويل النص إلى رقم نستخدم الدالة Number، وإذا لم يكن النص رقما صالحا تكون النتيجة NaN، ومعناها ليس رقما. وثمة طريقة تاريخية تستحق التذكر، وهي أن typeof عند تطبيقه على null يعيد الكلمة object. والسؤال هو: هل يطابق النوع ما يتوقعه السطر التالي من الكود؟ فمثلا `typeof quantity` يعيد الكلمة string عندما تأتي الكمية من نموذج.

مثال: ينظر أمين الصندوق إلى الورقة النقدية قبل قبولها، ولا يفترض أنها سليمة. والنظرة تستغرق ثانية وتمنع خطأ مكلفا. ودون هذه النظرة تدخل ورقة مزورة إلى الدرج، ولا تظهر الخسارة إلا في نهاية اليوم. وفحص نوع القيمة قبل استخدامها يعمل بالطريقة نفسها: فسؤال سريع واحد يمنع نتيجة خاطئة من السفر عبر البرنامج كله.

في العمل: تطبع رنا typeof quantity فترى الكلمة string. وتحول القيمة بالدالة Number، وتتأكد أن النتيجة ليست NaN، ولا تجمعها إلى الكعكات الموجودة في السلة إلا بعد ذلك. فيظهر العداد الآن خمسة، ويتلقى الزائر الذي يكتب حروفا رسالة واضحة بدلا من صفحة معطلة.

الخاتمة

يظهر عداد رنا الآن الرقم الصحيح ويقرأ الكود بوضوح، بفضل أربع خطوات. فالتصريح يسأل هل ستتغير القيمة لاحقا، والتسمية تسأل هل يفهم زميل الاسم. والنوع يسأل أي نوع من القيم هذه، والفحص يسأل هل يطابق النوع ما يتوقعه السطر التالي. وتفيد هذه العادات في كل وحدة لاحقة من السلسلة، لأن الدوال والمصفوفات والنماذج تعتمد كلها على قيم لها أسماء واضحة وأنواع معروفة.

المصادر

• شبكة مطوري موزيلا (MDN): أنواع البيانات وبنى البيانات في جافاسكربت – https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures
• شبكة مطوري موزيلا (MDN): تعليمة const – https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/const
• شبكة مطوري موزيلا (MDN): العامل typeof – https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/typeof
• شبكة مطوري موزيلا (MDN): القواعد والأنواع (أسماء المتغيرات والتصريح) – https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Grammar_and_types

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