عشر قواعد رئيسية لكي تبرمج مثل أفضل المبرمجين في ناسا NASA!



شورت بايتس **Short Bytes**: هل تعلم كيف يقوم كبار المبرمجين في ناسا بكتابة تعليمات مصيرية للمهمات في ناسا **NASA**؟

لجعل هذه التعليمات واضحة وآمنة وأسهل للفهم، قام مختبر جيت بروبولجن **Jet Propulsion Laboratory** التابع لناسا بوضع عشر قواعد من أجل تطوير البرامج.

يملك المطورون في ناسا واحدة من أكثر الوظائف تحدياً في عالم البرمجة، فهم يكتبون الشيفرة البرمجية ويطورون تطبيقات حساسة للمهام آخذين السلامة اعتبارهم الأول، ومن المهم في هذه الحالة اتباع بعض الإرشادات المصيرية، حيث تغطي هذه القواعد محاور مختلفة من عملية تطوير البرمجيات، مثل كيف يجب أن يكتب البرنامج؟ وأي خصائص للغة البرمجة يجب أن يتم استخدامها؟

وعلى الرغم من صعوبة الإجماع على قواعد موحدة للبرمجة الجيدة، فإن مختبر جت بروبولجن **JPL** التابع لناسا، يتبع مجموعة من قواعد البرمجة والمسماة "قوة القواعد العشر لتطوير برمجيات هامة وآمنة".

يركز الدليل بشكل رئيسي على البرمجيات المكتوبة بلغة **C** الخاصة بالبرمجة، وذلك نظراً للخبرة الطويلة لمنظمة **JPL** مع هذه اللغة. ولكن ومع ذلك، يمكن تطبيق هذه القواعد بسهولة مع لغات برمجة أخرى أيضاً.

تم تنظيم هذه القواعد من قبل رئيس العلماء في **JPL** جيرارد جي. هولزمان **Gerard J. Holzmann**، وتركز هذه القواعد الصارمة على الأمان والحماية.

قواعد ناسا العشر لكتابة برمجيات حساسة للمهام

---

1. تقييد كامل السطور البرمجية بأسلوب وانتقال بسيط – لا تستخدم تعليمات القفز **goto** أو **setjmp** أو **longjmp** أو النداء (التكرار) المباشر أو غير المباشر.

2. يجب أن يكون لجميع الحلقات حدود عليا ثابتة. حيث سيكون من الممكن بسهولة لأداة التحقق إثبات بشكل ستاتيكي فيما إذا كان ذلك الحد الأعلى الموجود في عدد التكرار للحلقة لا يمكن تنفيذه، وإذا لم يكن بالإمكان التحقق من ذلك الحد ستاتيكياً، فإن تلك القاعدة ستخرق.

3. لا تستخدم توزيع الذاكرة الديناميكي بعد التهيئة.

4. لا يجوز لأي تابع أن يكون أطول مما يمكن طباعته على وجه واحد للورقة، وذلك باستخدام مقياس مرجعي موحد، وباستخدام سطر واحد لكل عبارة وسطر واحد لكل تصريح. عادةً يعني ذلك عدم تجاوز **60** سطراً برمجياً لكل تابع.

5. يجب أن يكون متوسط كثافة التأكيد **Assertion** في البرنامج مساوياً على الأقل لتأكيدين لكل تابع. يتم استخدام التأكيدات لاختبار وجود أي حالات شاذة، والتي يجب ألا تحصل عند التنفيذ الفعلي، كما يجب أن تكون هذه التأكيدات دائماً خالية من الآثار الجانبية ومعرّفة كاختبارات بوليانية (منطقية: أصفار وواحدات) .**Boolean**. عندما يفشل التأكيد، يجب اتخاذ تصرف واضح للخروج منه، كأن يتم إرجاع الحالة الخاطئة إلى مستدعي التابع الذي يقوم بتنفيذ هذا التأكيد الفاشل. أي تأكيد يمكن للتحقق الثابت أن يثبت أنه لن يفشل أو لن يحمل أي خروقات لهذه القاعدة (أي أنه من غير الممكن تحقيق القاعدة عن طريق إضافة تعليمات **assert (true)**.

6. يجب أن يكون التصريح عن عناصر البيانات بأقصر طريقة ممكنة.

7. يجب أن يتم التحقق من القيمة التي تعيدها التوابع التي تعيد قيما **non-void functions**، وذلك عن طريق كل تابع مُستدعي، كما يجب التحقق من صحة القيم المُدخلة لكل تابع.

8. يجب أن يُحدد استخدام مرحلة قبل المعالجة لتضمين ملفات العناوين **Header Files** وتعاريف الماكرو **macro definitions** البسيطة. لا يسمح بلصق الرموز ولوائح النقاش المتغيرة **elippses** واستدعاءات الماكرو المتكررة، بل يجب أن يتم توسيع جميع تعليمات الماكرو لتكون كوحدة مركبة متكاملة، ولا يفضل أيضاً استخدام تعليمات البناء الشرطي، ولكن لا يمكن دائماً تجنب استخدامها، وهذا يعني أنه من المفضل وجود تبرير لوجود القليل من تعليمة بناء شرطي واحدة أو اثنتين، حتى في عمليات تطوير البرامج الكبيرة.

بالذهاب أبعد من المقاييس المتداولة والتي تتجنب التضمين المتكرر لملف العنوان ذاته، حيث يجب تأشير كل استخدام له بمدقق معتمد على أداة، ويجب تعديلها في السطور البرمجية.

9. يفضل عدم استخدام المؤشرات، وبالتحديد عدم استخدام أكثر من مرحلة واحدة من إلغاء التأشير، فإن عمليات إلغاء التأشير قد لا تكون مخفية في تعاريف الماكرو أو بداخل التصريح **typedef**، كما أنه لا يسمح بمؤشرات التوابع.

10. يجب ترجمة كل السطور البرمجية منذ اليوم الأول للتطوير مع كل تحذيرات المترجم **compiler** التي يتم تفعيلها في إعدادات المترجم. يجب أن تتم ترجمة كل السطور البرمجية دون وجود أي تحذيرات. ويجب أن يتم تدقيق البرنامج يومياً مرة واحدة على الأقل، ويفضل أن يكون أكثر من مرة، عن طريق محلل برمجيات بمصدر ثابت ومثالي، ويجب أن يجتاز البرنامج الاختبار دون أي تحذيرات.

وفيما يخص هذه القواعد، تقول ناسا

تعمل هذه القواعد كحزام الأمان في سيارتك: في البداية، قد تكون غير مريحة بعض الشيء، لكن بعد فترة يصبح استخدامها لا إرادياً، ولا نستطيع تخيل العمل من دونها.

- التاريخ: 2016-12-25
- التصنيف: تكنولوجيا

#برمجة #ناسا

المصطلحات

- **الأيونات أو الشوارد (Ions)**: الأيون أو الشاردة هو عبارة عن ذرة تم تجريدها من الكترون أو أكثر، مما يُعطيها شحنة موجبة.وتسمى أيوناً موجباً، وقد تكون ذرة اكتسبت الكتروناً أو أكثر فتصبح ذات شحنة سالبة وتسمى أيوناً سالباً

المصادر

- fossbytes

المساهمون

- ترجمة
  - رامي قباني
- مُراجعة
  - مخائيل سلامه
- تحرير
  - روان زيدان
- تصميم
  - نور سلمان

- نشر
  - عصام عباس