

تعريف ببعض أدوات المبرمج في عالم البرمجيات الحرة المفتوحة المصدر

خالد الشمعة k.el-shamaa@cgiar.org

<http://www.ar-php.org>

مقدمة:

لقد إنطلق مشروع PHP واللغة العربية في بدايات العام 2006، وكانت الفكرة تقضي الاستفادة من أوقات الفراغ في بناء مكتبة برمجية حرة مفتوحة المصدر تهدف إلى خدمة لغتنا العربية من خلال توفير مجموعة متنوعة من الوظائف المخصصة لها، وقد منحتنا طبيعة المشروع أفضلية نسبية على غيرنا من خبراء لغة PHP العاملين في مجال البرامج الحرة المفتوحة المصدر كون أهل مكة أدرى بشعابها، وهكذا توالى إنتاجات المشروع التي لاقت استحسان مجتمع المبرمجين على بساطة بعضها، حتى أنه تم تكريم سبع منها على تميزه من قبل إدارة وزوار موقع www.phpclasses.org، وكان نصيب أربع منها التتويج كأفضل منتجات لأشهر التي نشرت فيها.

ساهمت ردة الفعل الإيجابية هذه ببث مزيد من روح الحماسة في المشروع، فأخذ يستهلك معظم أوقات الفراغ التي امتلكتها، وأخذ يتحول من تجربة لاهية إلى رسالة سامية، فتوالى منتجاته التي تضمنت على سبيل المثال لا الحصر البحث بدلالة ساق الكلمة والتلخيص الآلي والتحويل ما بين التاريخين الهجري والميلادي والترجمة اللفظية للكلمات ما بين الأبجدية العربية واللاتينية. للاطلاع على اللائحة الكاملة مما هو متوافر من مكتبات يمكنكم زيارة الموقع

<http://www.ar-php.com>

مع نهاية العام 2007 وصل هذا المشروع مرحلة من النضج وكما من المنتجات استدعت الانتقال به خطوة جديدة إلى الأمام نحو مزيد من الاحتراف والمعيارية في عالم البرمجيات الحرة المفتوحة المصدر، وهكذا انطلقنا بداية العام 2008 بالعمل على جمع برامج المشروع المتفرقة في مكتبة واحدة مستفيدين بشكل عميق من المزايا الأصلية للبرمجة الغرضية التوجه التي بات الإصدار الخامس من لغة PHP يوفرها، حيث انصب اهتمامنا على تحسين جودة شيفرتنا المصدرية وطرائقنا الخوارزمية والتعليمات البرمجية التي كنا ننتقيها بعناية لخدمة هدف زيادة سرعة التنفيذ وخفض مساحة الذاكرة المستهلكة، وهي عملية لا تزال مستمرة حتى الآن.

لم تقتصر نشاطات الإنطلاقة الجديدة للمشروع على ذلك فحسب، بل تعدتها إلى طيف من الخدمات الداعمة المحيطة به والتي تكمل بدورها صورته وتعزز مكانته ليقف على قدم المساواة بين غيره من البرمجيات الحرة المفتوحة المصدر كإضافة جديدة صغيرة لكنها تتبع ذات الإجراءات والمعايير القياسية التي يتبعها الكبار من محترفي هذا

القطاع من تقنية المعلومات، فمن إدارة أفضل للشيفرة المصدرية التي باتت تنشر الآن على موقع sourceforge.net، إلى وضع آلية لضبط الجودة من خلال تطبيق عملية فحص آلية مؤتمتة تختبر كافة مرافق المكتبة عقب إجراء أي تعديل عليها وذلك قبل نشر النسخة المعدلة من جديد، هذا عدى عن إضافة توثيق متكامل للشيفرة المصدرية بصيغة معيارية، كل ذلك والمزيد تم نشره في موقع جديد هو <http://www.ar-php.org>. لقد كان الدافع من وراء ذلك هو السعي الدؤوب إلى إضافة هذا العمل إلى مكتبة PEAR القياسية الخاصة بلغة PHP، لقد كان هذا الهدف يبدو بعيد المنال مع بدايات هذا العام، لكننا الآن قاب قوسين أو أدنى من إتمام إجراءات التسجيل والإضافة بعد الوفاء بجميع الشروط والمتطلبات التي تحتاج إليها، وقد اتسمت الرحلة ما بين هذين النقيضين بالكثير من الجهد وترافقت بتعلم العديد من التقنيات الجديدة واكتساب المزيد من الخبرات التي ساهمت في تحقيق هذا الحلم وجعله واقعا.

سأعمل من خلال ورقة البحث هذه على استعراض بعض من الأدوات التي ساعدت على تحقيق هذا الإنجاز، حيث أمل من خلال مشاركتكم هذه التجربة أن أساعد في تمهيد الطريق أمام غيري من المبرمجين المهتمين بالمشاركة الفاعلة في عالم البرمجيات الحرة المفتوحة المصدر، وذلك من خلال تعريفهم ببعض الأدوات المفيدة، حيث تدرج أهمية هذه الأدوات في إطار ممارسة إحدى الحقوق التي يضمنها ترخيص البرمجيات الحرة المفتوحة المصدر للمستخدم وهو الحق في الوصول إلى الشيفرة المصدرية والتعديل عليها، وهو ما يعطي أي مستخدم لهذا النوع من البرمجيات الكثير من المرونة والاستقلالية في تخصيص وضبط هذه البرمجيات بما يلائم احتياجاته وخاصة في حالة الشركات والمؤسسات التي يتوفر لها الكوادر المؤهلة والإمكانات المادية، ولا تشذ تطبيقات الويب المفتوحة المصدر عن هذه القاعدة.

نظام CVS لإدارة الشيفرة المصدرية:

إن كنت ممن إعتادوا على تنزيل البرامج الحرة والمفتوحة المصدر من هنا أو هناك على شبكة الويب، فلا بد أنك قد تعرفت إلى موقع Sourceforge.net والذي يعد واحدا من أكبر مكانز البرمجيات المفتوحة المصدر وأكثرها مصداقية، وهو الذي يحفظ الشيفرة البرمجية للآلاف من تلك المشاريع (حاليا هناك أكثر من 100,000 مشروع منشور ضمن هذا الموقع وأكثر من 1,000,000 مطور مسجل)، وتدرج تلك المشاريع تحت العشرات من الفئات والأنواع ولغات البرمجة وبيئات العمل وأنظمة التشغيل.

إن أسباب شعبية هذا الموقع وإعتماده من قبل العديد من المشاريع المرموقة لا تنحصر في توفير المكان الذي يستضيف ملفات المشروع وشيفرته المصدرية بشكل مجاني (على الرغم من أهمية ذلك)، إضافة إلى مجموعة الخدمات الأخرى من قبيل منتدى الدعم التقني ونظام إرسال تقارير الأخطاء ومتابعتها هذا عدى عن توفير الإحصائيات المختلفة وإمكانية إقتراح مزايا وخصائص جديدة من قبل مستثمري المشروع البرمجي، والكثير غيرها

من الخدمات اللوجستية التي تغطي معظم طيف الخدمات التي يمكن أن يحتاج إليها أي مشروع برمجي للتفاعل مع مستخدميه.

لكن الميزة الأكثر جوهرية وأهمية خصوصا لفريق العمل الذي يقوم بتطوير ذلك المشروع أو البرمجية هو توفير مخدّم (CVS (Concurrent Versioning System)، وهو عبارة عن نظام مفتوح المصدر مخصص للتحكم بالإصدارات، بمعنى مراقبة كافة التغييرات التي تطرأ على ملفات الشيفرة المصدرية للبرمجية خلال مراحل تطوير المشروع، ويتيح الفرصة للعديد من المطورين للعمل معا حتى وإن كانوا متواجدين في مواقع متباعدة مكانيا أو مختلفة من حيث الفوارق الزمنية.

لذا إن كنت مبرمجا تطمح للمساهمة يوما ما في أي مشروع برمجي مفتوح المصدر، لا بد لك من أن تتقن طريقة التعامل مع مخدّم CVS لتكون قادرا على استخراج الشيفرة المصدرية والحصول عليها، لتقوم بعد ذلك بإضافة تعديلاتك إليها، ومن ثم إيداع تلك التعديلات التي أضفتها مجددا في مكنز الشيفرة المصدرية الأصلية للمشروع، لتصبح جزءا منه وتكون بالتالي متاحة للآخرين.

إن توفر برمجية زبون CVS هو أمر ضروري للمطور حتى يكون قادرا على التعامل مع مخدّم CVS سواء ذلك الموجود على موقع Sourceforge.net أو غيره من المواقع التي توفر مثل هذه الخدمة، أو حتى على شبكتك الداخلية في حال توفر مخدّم CVS محلي. يعتبر برنامج TortoiseCVS واحدا من أشهر برامج زبون CVS وأكثرها بساطة وسهولة وتفاعلية في بيئة نظام التشغيل Windows، إضافة إلى كونه مجاني، حيث يمكنك الحصول عليه من الموقع: <http://www.tortoise cvs.org> (بالنسبة لمخدّم CVS يمكنك استخدام CVSNT وهو مخدّم مجاني متاح على أنظمة التشغيل Windows، ويمكن الحصول عليه من الموقع <http://www.cvsnt.org>)

تتبع سهولة هذا البرنامج من حقيقة إندماجه في بيئة Windows عوضا عن عمله ككيان أو برمجية مستقلة، وهو ما يجعل التعامل معه سهلا حتى بالنسبة للمستخدمين الجدد، حيث يمكنك الوصول إلى كافة الوظائف من خلال النقر على الزر الأيمن للفأرة، وهو يقوم من خلال تلك القائمة بتوفير وصول مباشر لعمليات CVS الأكثر إنتشارا فقط وذلك لتحاكي إرباك المستخدم.

فيما يلي سوف نستعرض سيناريو متكامل ليوم عمل نتفاعل خلاله مع مختلف وظائف مخدّم CVS الشائعة. عند بدء العمل، عليك الحصول على نسخة كاملة من الشيفرة المصدرية للمشروع (checkout) على حاسوبك الخاص من مخدّم CVS (أو ما يدعى بصندوق الرمل الخاص بك). بعد ذلك تقوم ببساطة وبإسلوبك المعتاد وأدواتك الخاصة بالعمل على الملفات التي ترغب بتعديلها، أو تضيف ملفات جديدة أو حتى تحذف أخرى موجودة، وبعد الإنتهاء من تنفيذ تعديلاتك وإختبار صحتها، عليك إعادتها إلى المكنز من جديد باستخدام تعليمة الإيداع commit لتنفيذ عملية

check in. فإن صادف وأن قام أحدهم بتعديل ذات الملف الذي عملت عليه، حينها ستفشل عملية الإيداع commit التي تحاول تنفيذها، حينها يتوجب عليك تنفيذ تعليمة تحديث update لكامل الشيفرة المصدرية التي لديك في صندوق الرمل بالاعتماد على الإصدار الأحدث التي باتت متوفرة على مكنز مخدم CVS، حيث تقوم هذه التعليمة بدمج تعديلات المطورين الآخرين بشكل آلي في نسختك من ملفات الشيفرة المصدرية ضمن صندوق الرمل الخاص بك.

في بعض الأحيان لا يمكن إنجاز تلك المهمة بشكل آلي تماماً، كأن يقوم كلاهما بتعديل ذات السطر البرمجي، وهو ما يدعى بالتضارب conflict. إن احتمال حدوث مثل هذه التضاربات في الحياة العملية هو أقل بكثير مما تتصور، لكنها إن حدثت، فسيقوم نظام CVS بوضع كلا الإصدارين من الشيفرة المتضاربة ضمن ذات الملف، مع علامة واضحة تفرق فيما بينهما، وبعدها عليك تعديل الملف بالشكل المناسب يدويا لتحل هذا التعارض وتحافظ على المنطق البرمجي سليماً، وتزيل العلامات التي استخدمت للإشارة إلى مكان التعارض من قبل نظام CVS، بعد ذلك تستطيع إعادة محاولة عملية الإيداع commit من جديد، والتي ستجح في هذه المرة.

تتمتع طريقة العمل هذه بالكثير من المزايا، فلكل مطور صندوق الرمل الخاص به، وبالتالي تكون التعديلات التي يقوم بها المطورون الآخرون معزولة عما تقوم به أنت من تعديلات، إلى أن ترغب في إيداع تعديلاتك بعد الإنتهاء منها. وهذا ما يلغي عنق الزجاجة الذي كان يمنع الآخرين من القيام بأي شيء لأن أحد ما يعمل على تنفيذ تعديل معين في قسم ما من المشروع خشية حذف تعديلات الآخرين بشكل عرضي. وهكذا أصبح باستطاعة أي مطور العمل على الملفات دون الحاجة إلى إتصال مباشر مع المخدم، وباتت عملية الإتصال تنحصر عند الرغبة في إيداع التعديلات أو طلب تحديث النسخة المحلية للشيفرة المصدرية من المكنز.

إن الوظيفة الأساسية لنظام CVS هي السماح لعدة مطورين بالعمل بشكل مستقل على ذات مجموعة الملفات، ليتم لاحقاً دمج تعديلاتهم في المكنز المركزي، وبالتالي يعد هذا النظام أحد المفاتيح الأساسية لنجاح عمل الفرق البرمجية وواحداً من المعايير العالمية لإدارتها، ولا يقتصر دور نظام CVS على توفير مكنز مركزي يتيح توزيع نسخ عن الشيفرة المصدرية لأعضاء فريق العمل، ومن ثم دمج التعديلات التي يجرونها في ذلك المكنز مجدداً، بل يسمح لك أيضاً بتتبع من قام بإجراء التعديلات، وأين تمت تلك التعديلات، ومتى تم إجراؤها. كذلك يمكنك معرفة لماذا أجريت تلك التعديلات في حال إلترم المبرمجون الذين يستثمرون هذا النظام بإضافة تعليقات مفيدة مع كل عملية إيداع يقومون بها. وهذه ميزات هامة جداً، خصوصاً لمن يقومون بإدارة الفرق البرمجية ومراقبة عمل أفرادها.

من جهة ثانية يعتبر المكنز (والذي يحتفظ بكافة التعديلات التي أجريت على الشيفرة المصدرية للبرمجية خلال كامل تاريخ تطورها) دليلاً قوياً يمكن اللجوء إليه لإثبات الجهة التي قامت بتطوير البرمجية عند أي تحكيم يجري بهذا الخصوص.

أما إن لم يكن لديك فريق عمل يقوم بتطوير برمجية ما، وأردت أن تتعامل مع نظام CVS على أنه مجرد بديل لعملية النسخ الاحتياطي Backup، فستحصل على جملة من المزايا الإضافية منها أن نظام CVS يقوم بحفظ كافة الإصدارات بدءاً من تاريخ إستثمار النظام، على عكس أسلوب النسخ الاحتياطي التقليدي، حيث نتخلص عادة من النسخ القديمة بعد فترة. كذلك يستطيع نظام CVS التراجع عن أي من التعديلات التي تمت خلال أي لحظة زمنية سابقة بسهولة ويسر، متيحاً لنا إمكانية استرجاع حالة الملفات كما كانت في أي تاريخ سابق.

يقوم نظام CVS بحفظ الملفات ضمن مكنز مركزي، لكنه بحسب طريقة عمله يقوم كذلك بإنشاء نسخ احتياطية موزعة على أجهزة المبرمجين المحلية، وهي ذاتها صناديق الرمل التي سبق وأن استخرجوها من المكنز، لذا فإن إدارة المخاطر هنا أفضل من حالة النسخ الاحتياطية التقليدية. هذا عدى عن الإقتصاد في استخدام عرض نطاق الحزمة الشبكي المتوفر ومساحة التخزين المستخدمة على المخدم، وذلك لأن النظام يقوم بحفظ التعديلات فقط نسبة إلى الإصدار السابق.

على كل حال، ومع كل هذه القيمة المضافة التي يقدمها لنا توفر مخدم CVS محلي، إلا أنه تجب الإشارة إلى ما لا يقوم به نظام CVS، منعا للالتباس والخطأ، حيث لا يعتبر هذا النظام وسيلة للإتصال فيما بين أعضاء الفريق البرمجي الواحد، وهو ليس بديلاً عن العمل الجماعي، كما لا يعتبر نظاماً خاصاً بتتبع الأخطاء ومتابعة عملية إصلاحها، ولا يوفر أي أدوات لفحص الشيفرة البرمجية واختبارها.

لمزيد من المعلومات والتفاصيل يرجى العودة إلى مقالة "أدوات زبون CVS" التي نشرت في مجلة المعلوماتية، العدد 26، أبريل - أيار 2008، الصفحات 30-33. وكذلك مقالة "تنصيب وإعداد مخدم CVS" المنشورة أيضاً في مجلة المعلوماتية، العدد 28، يونيو - حزيران 2008، الصفحات 50-53.

إطار عمل PHPUnit لبناء الاختبارات المؤتمتة:

إن كل من مارس البرمجة يعلم تماماً أن الوقوع في الأخطاء هو أمر لا مفر منه، لكن على المبرمج المحترف أن يخطط مسبقاً لتفحص الأخطاء والمواظبة على اختبار ما يكتبه من شيفرات برمجية سعياً وراء اصطيد مبكر للأخطاء مما يعزز من فرص اكتشافها ويقلل من كلفة إصلاحها، وتعتبر عمليات الاختبار تلك واحداً من الأركان الأساسية لعمليات مراقبة وضبط الجودة التي لا غنى عنها في عالم الاحتراف.

عادة ما يستخدم المبرمجون تعليمات مثل `echo()` و `var_dump()` لطباعة بعض القيم أو عرض شيء من المعطيات المرئية للتأكد من تفاصيل عمل القطعة البرمجية وحسن سيرها، أو لتشخيص أسباب سلوك غريب خاطئ لها، لكن مثل هكذا طرق عادة ما تكون قاصرة على مرحلة التطوير ومستهلكة للوقت، هذا عدى عن كونها حكرًا على من غاص بين ثنايا الأسطر البرمجية وعمل على فك ألغازها وغموضها.

في هذا الميدان تأتي أداة PHPUnit لتقدم إطار عمل بسيط وسهل الاستخدام يساعد المبرمج على بناء وتنفيذ عمليات الفحص والاختبار المؤتمتة للوحدات البرمجية المبنية وفق مبادئ البرمجة الغرضية التوجه وذلك على مستوى الأصناف Classes وطرائقها Methods. فبمجرد وضع تصميم الصنف الذي ستعمل على تطويره تصبح قادرا على بناء الاختبار المؤتمت الكفيل بتفحصه. التصميم هنا يأتي بمعنى تحديد مجموعة خصائص الصنف وطرائقه مع توصيف كامل لوسطاء الدخل وما يفترض أن يقابلها من قيم خرج لكل واحدة من الطرائق المتوفرة للصنف تبعا للحالة التي يكون عليها الكائن (أي قيم خصائصه). كما تلاحظ فإن بناء الاختبار المؤتمت يمكن له أن يسبق كتابة الشيفرة البرمجية للصنف ذاته، وهو يساعد على استيضاح ما يطلب تنفيذه تماما، ويستخدم لاحقا كأحدى أدوات التحقق من استيفاء الشروط والمواصفات المتفق عليها.

إن برنامج PHPUnit هو فرد من عائلة مكتبة PEAR، فإن لم يكن متوفر على المخدم الذي تستخدمه لعملية التطوير، فكل ما عليك القيام به هو طلب تنصيبه ببساطة من خلال تنفيذ التعليمتين التاليتين في سطر الأوامر:

```
pear channel-discover pear.phpunit.de
pear install phpunit/PHPUnit
```

إن إلحاق شيفرات الاختبارات المؤتمتة بأي برمجية حرة مفتوحة المصدر تمكن المستخدم من التحقق من صحة عمل تلك البرمجية على منصته ويمنحه مزيدا من الثقة بها كون فريق العمل المعني بتطويرها يتبع إجراءات ضمان الجودة المتعارف عليها. إن فائدة توفر مثل هذه الاختبارات المؤتمتة يساعد أيضا كل من يحاول تعديل أو تطوير برمجية ما على التأكد من أن التغييرات التي قام بها لم تتسبب في ظهور أي خلل في بقية أرجاء البرمجية.

قد تبدو عملية تصميم وبناء الاختبارات المؤتمتة مستهلكة للوقت، لكن هذا الانطباع خاطئ، فبناء اختبار جديد باستخدام إطار عمل PHPUnit لا يستغرق أكثر من بضع عشرات من الدقائق، كما أن تنفيذ الاختبار ذاته يتم في ثوان. يوضح المثال التالي جزءا من الاختبار المؤتمت المرفق مع ملفات مشروع PHP واللغة العربية:

المصادر وما إلى ذلك، فيما تمتاز تسمية كافة حالات الاختبار بأنها تبدأ حصراً بالكلمة test على أن يتلوها ما يوضح طبيعة حالة الاختبار هذه، وكل من حالات الاختبار هذه (أو لنقل الطرائق) تتضمن الطريقة الموروثة assertEquals لإتمام عملية الفحص عقب تهيئة الدخل وتمريضه للحصول على الخرج ومقارنته بما يتوقع الحصول عليه.

إن تنفيذ عملية الاختبار تتم ببساطة من خلال سطر الأوامر من خلال تنفيذ الأمر:

```
phpunit testName
```

حيث يتم تمثيل كل حالة اختبار ناجحة بطباعة الرمز النقطة ضمن الخرج، فيما يستخدم الرمز F للإشارة إلى الاختبارات التي فشلت بسبب عدم تطابق الخرج الناتج مع الخرج المتوقع مع رسالة توضح طبيعة الاختلاف ما بين القيمتين من أجل كل حالة اختبار فاشلة.

يوضح الشكل التالي حالة تنفيذ اختبار نموذجي صحيح بالكامل:

```
C:\xampp\htdocs\Arabic>phpunit ArabicTest.php
PHPUnit 3.2.16 by Sebastian Bergmann.

.....

Time: 0 seconds

OK (36 tests)
```

يوضح الشكل التالي حالة تنفيذ ذات الاختبار مع وجود أخطاء (الخطأ الأول سببه ظهور الحرف E عوض الحرف A فيما الخطأ الثاني سببه إعادة متحول منطقي غير مطابق لما هو متوقع):

```
C:\xampp\htdocs\Arabic>phpunit ArabicTest.php
PHPUnit 3.2.16 by Sebastian Bergmann.

.....FF.....

Time: 0 seconds

There were 2 failures:

1) test_arabic_english_transliteration(ArabicTest)
Failed asserting that two strings are equal.
expected string < Khalid El-Sham'ah>
difference < x>
got string < Khalid Al-Sham'ah>
C:\xampp\htdocs\Arabic\ArabicTest.php:146

2) test_arabic_gender_male(ArabicTest)
Failed asserting that <boolean:false> matches expected value
C:\xampp\htdocs\Arabic\ArabicTest.php:156

FAILURES!
Tests: 36, Failures: 2.
```


برنامج phpDocumentor لتوثيق الشيفرة البرمجية ألياً:

يعد التوثيق المتكامل والوافي واحداً من أهم سمات الاحترافية في العمل البرمجي، لكنه من جهة أخرى يصنف على أنه واحد من المهام التي تصيب معظم المبرمجين بالملل كونه يستهلك الكثير من الوقت الذي يصرف بعيداً عن محبوبتهم البرمجة، لذا وجب على المبرمج المحترف أن يجد حلاً وسطاً ما بين شغف البرمجة ورصانة التوثيق، وهنا يأتي دور برنامج phpDocumentor الذي يهدف إلى مساعدة المبرمجين على توليد توثيق معياري بشكل شبه آلي إنطلاقاً مما يكتبونه من ملاحظات وتعليقات بين ثنايا شيفراتهم المصدرية شريطة إتباعهم طريقة معينة في كتابة تلك الملاحظات والتعليقات (سنأتي على ذكر تركيبها وصيغتها بعد قليل)، إضافة إلى كون هذا البرنامج يقوم بشكل آلي بتفحص شيفرتك المصدرية وتضمين ما يتوفر لديه من تفاصيل حول الثوابت والتوابع والأصناف وطرائقها وسواها من بنى جوهرية تحتاج إلى توضيح أو إشارة ضمن التوثيق.

إذن لننتقل بداية على أن الملاحظات والتعليقات التي تسرد ضمن الشيفرة المصدرية هامة وضرورية ليس فقط لمن هو منخرط في إطار فريق عمل، بل هي هامة لك حتى وإن كنت تعمل بشكل مستقل على برنامج تعتبره بسيطاً وصغيراً، فالأمور لن تبقى كذلك بعد عام أو أكثر، وسيطلب الأمر منك أن تغوص مجدداً في شيفرتك المصدرية التي سبق وأن وضعتها أنت نفسك قبل حين محاولاً تذكر واستكشاف طريقة عمل ما قد صنعه يداك قبلاً، فكيف سيكون الحال مع الآخرين؟ لذا فإن وضع التعليقات الوافية ضمن الشيفرة المصدرية ذاتها على الأقل يعدّ بحق استثماراً رابحاً على المدى البعيد، ويختصر عليك وعلى الآخرين الكثير من الوقت والجهد عند الحاجة إلى التعديل على تلك الشيفرة المصدرية، وقد ساهمت مفاهيم البرمجة الغرضية التوجه من خلال ميزة التغليف في ضبط وعزل الكينونات البرمجية على شكل أصناف تمتلك خصائص وطرائق، فتتخصص مهمة التوثيق بإضافة التعليقات المناسبة لكل صنف وطريقة وخاصة موضحاً دورها ودخلها وخرجها ووجهة استخدامها بحيث تستطيع أن تترفع عن ما هو دون ذلك من تفاصيل داخل النصوص البرمجية ذاتها.

```
/**
 * Arabic Date
 *
 * PHP class for Arabic and Islamic customization of PHP date function. It
 * can convert UNIX timestamp into string in Arabic as well as convert it
 * into Hijri calendar
 *
 * @copyright Khaled Al-Shamaa 2008
 * @link http://www.ar-php.org
 * @author Khaled Al-Shamaa <khaled.alshamaa@gmail.com>
 * @package Arabic
 */
```

```
class ArDate
{
```

يوضح المثال السابق حالة نموذجية لتعليق يسبق إحدى الطرائق الواردة في مشروع PHP واللغة العربية يوضح فيه بنية مقطع التعليق الذي يجب أن يبدأ بسطر **/ وينتهي بسطر */ على أن تسبق كل سطر من أسطر مقطع التعليق رمز النجمة *

يمكن أن يتألف مقطع التعليق من سطر أول يمثل شرحا مقتضبا أو عنوانا يصف ما يتلوه من كينونة برمجية سواء كانت تابعا أو متحولا أو صنفا الخ... يمكن أن يتلوه هذا العنوان شرح أطول وأكثر تفصيلا على أن يفصل بينهما سطر فارغ (دون أن ننسى رمز النجمة الملزم في كل أسطر مقطع التعليق حتى ولو كان فارغا)، يمكن لهذا الشرح المطول أن يقسم على عدة فقرات على أن يفصل بين كل منها سطر فارغ وحينها يعامل مضمون تلك الفقرات على أنه نص صرف خال من التنسيق، إلا أنه من الممكن استخدام بعض معرفات HTML ضمن فقرات الشرح المطول هذا مثل و <i> و و و و
 و <p> شريطة أن يتم استخدامها من بداية الشرح المطول لتكون هذه البداية على الشكل التالي <p>*, طبعا لن تظهر هذه التنسيقات بالهيئة المتوقعة منها إلا حينما يتم تصديرها لاحقا إلى صيغ أخرى مثل ملفات التوثيق بصيغة HTML أو CHM أو حتى PDF (وهو الدور المنوط ببرنامج phpDocumentor).

من جهة أخرى تتضمن فقرات التعليقات هذه إضافة للعناوين والشرح المفصل مجموعة كبيرة من المعرفات المحددة المعاني والتي يمكنها توصيف القطعة البرمجية التي تتلوهما بشكل أكثر هيكلية وتوحيدا، حيث يشترط أن يذكر أي من هذه المعرفات بسطر مستقل ويستخدم له تسمية محددة متفق عليها على أن تبدأ بالرمز @، نذكر فيما يلي بعضا من أهم تلك المعرفات وصيغ إيرادها ضمن كتلة التعليقات:

```
* @param type [$varname] description
```

ويذكر هذا السطر من أجل واحد من وسطاء الدخول للتابع أو الطريقة وبذات تسلسل ورودها مع ذكر نوع هذا الوسيط سواء كان عددا صحيحا أو سلسلة نصية أو قيمة منطقية أو حتى مصفوفة أو ما عداها، ثم يتم ذكر اسم هذا الوسيط (مع البادئة \$ كما هي العادة في لغة PHP) يتلوه شرح مبسط لمعنى هذا الوسيط وما يتوقع أن يوضع فيه. هناك أيضا المعرف:

```
* @return type description
```

وهو يضاف إن كان التابع أو الطريقة صاحبة التعليق تعيد قيمة ما لمن استدعاها، فيتم توضيح نوع القيمة المعادة مع سطر يشرح فحواها. هنالك أيضا الكثير من هذه المعرفات المفيدة التي يدل اسمها على دور كل منها، فهناك مثلا:

```
@copyright, @link, @author, @package
```

يوضح الشكل التالي مثالا آخر عن طريقة الاستخدام مع التوابع أو الطرائق:

```
/**
 * @return TRUE if success, or FALSE if fail
 * @param Integer $mode Output mode of date function where:
 *
 *         1) Hegri format (Islamic calendar)
 *         2) Arabic month names used in Middle East countries
 *         3) Arabic Transliteration of Gregorian month names
 *         4) Both of 2 and 3 formats together
 *         5) Libyan way
 * @desc Setting value for $mode scalar
 * @author Khaled Al-Shamaa <khaled.alshamaa@gmail.com>
 */
public function setMode($mode = 1)
{
    $flag = true;

    if (in_array($mode, array(1, 2, 3, 4, 5))) {
        $this->mode = $mode;
    } else {

        $flag = false;
    }

    return $flag;
}

/**
 * @return String Format Arabic date string according to given format string
 *         using the given integer timestamp or the current local
 *         time if no timestamp is given.
 * @param String $format Format string (same as PHP date function)
 * @param Integer $timestamp Unix timestamp (default is current local time)
 * @desc Format a local time/date in Arabic string
 * @author Khaled Al-Shamaa <khaled.alshamaa@gmail.com>
 */
public function date($format, $timestamp = time)
{
    if ($this->mode == 1) {
        $hj_txt_month[1] = 'أغرم';
        $hj_txt_month[2] = 'أصفر';
    }
}
```

إن توافر هذه الصيغ من التعليقات داخل شيفرتك المصدرية يجعل عملية التوليد الآلي للتوثيق سهلة المنال وبسيطة، فكل ما عليك القيام به هو استدعاء برنامج phpDocumentor من خلال سطر الأوامر (تذكر أن برنامج phpDocumentor هو في نهاية المطاف مكتوب بلغة PHP ذاتها، لذا على مسار مفسر لغة PHP أن يكون معرفا في بيئة مخدم التطوير لديك)، طبعا عليك تمرير مجموعة من الوسائط التي تحدد للبرنامج الملف أو المجلد الذي يتضمن الشيفرة المصدرية المراد توثيقها (f- يتلوه اسم الملف مع المسار المؤدي إليه أو d- يتلوه المسار المؤدي إلى المجلد الذي يتضمن برنامجك)، كذلك عليك تحديد صيغة الخرج التي تريد لملفات التوثيق أن تظهر بها (o- يتلوه التنسيق المرغوب)، فمثلا يمكنك استخدام HTML:Smarty:PHP لتحصل على توثيق HTML بصيغة تماثل ما

هو مستخدم في الموقع الرسمي للغة PHP، أو يمكنك استخدام PDF:default:default للحصول على ذلك التوثيق كملف PDF، أو حتى يمكنك طلب التنسيق CHM:default:default لتولد لاحقا ملف توثيق بصيغة CHM. هناك أيضا الكثير من الوسائط المفيدة الأخرى التي يمكن تمريرها إلى تعليمة سطر الأوامر للتحكم بخصائص التوثيق الناتج (مثلا t- لتحديد المجلد الذي ستوضع فيه ملفات التوثيق الناتجة، أو ti- التي يتلوها العنوان الأساس الذي نريد له أن يظهر في ترويسة كل صفحة من صفحات التوثيق).

ملاحظة: إن برنامج phpDocumentor هو فرد من عائلة مكتبة PEAR، فإن لم يكن متوفر على المخدم الذي تستخدمه لعملية التطوير، فكل ما عليك القيام به هو طلب تنصيبه من مكنز هذه المكتبة، وهو ما يتم ببساطة من خلال تنفيذ التعليمة التالية في سطر الأوامر:

```
pear install phpDocumentor
```

وهكذا أصبح بإمكانك الحصول على توثيق احترافي بهيئة الطلة باتباع خطوات يسيرة ودون أن تستهلك الكثير من الوقت، إنما تنحصر في مجموعة من الخطوات البسيطة والمعايير التي يجب إتباعها والتقيد بها خلال عملك.

الخلاصة والتوصيات:

إن التعريف بالأدوات المساعدة في كل من عمليات إدارة الشيفرة المصدرية وأتمتة الاختبارات وتوثيق الشيفرة المصدرية هي مسألة لا تقل أهمية عن إتقان لغة البرمجة المستخدمة ذاتها، فالاحتراف ومتطلبات العمل الجماعي ضمن الفرق تتطلب مهارات لا تنحصر بمجرد صقل موهبة التحليل والبرمجة. لذا أرى أن يشار إلى مثل هذه الأدوات ضمن السياق الملائم في المناهج التعليمية التي تؤهل كوادرننا من المبرمجين.