STUDIA PHILOLOGICA UNIVERSITATIS VELIKOTARNOVENSIS
VOL. 36/2 2017 VELIKO TARNOVO

Стефан СТЕФАНОВ

ВТУ „Св. св. Кирил и Методий“, България

ПРЕВОД НА ТЕКСТ, СЪДЪРЖАЩ ПРОГРАМЕН КОД

Stefan STEFANOV

“St Cyril and St. Methodius” University of Veliko Tarnovo, Bulgaria

TRANSLATING TEXTS CONTAINING PROGRAMMING CODE

The text discusses the growing occurrence of inclusion of programming code in translation projects and the challenges this poses to translators. A number of examples from practice are analysed identifying translatable and non-translatable content – variables from common programming languages and client-customised placeholders – emphasising the importance of preserving the code intact. Examples of some conventional localisation file types are adduced as well.

Ключови думи: превод, програмен код, променливи, контейнери, непреводимо съдържание, локализация

Keywords: translation, programming code, non-translatable content, placeholder, localisation, localization

Преобладаващата част от писмената комуникация в наши дни се осъществява с помощта на електронни средства, което ще рече, че четем и пишем предимно на екрана на компютър или мобилно устройство. За да може да достигне текстът до своя получател и най-вече за да бъде персонализиран, той в голяма част от случаите представлява комбинация от човешки и компютърен език – текст, смесен с програмен код. Когато възникне необходимостта от превеждане на такъв тип дигитална комуникация, от преводача вече се изисква не просто да притежава необходимата компетенция в изходния и целевия език, но и да умее да борави с програмен код до такава степен, че в процеса на превода да идентифицира преводимите и непреводимите структури и да пренася последните в преводния текст по такъв начин, че съдържащият се в тях код да продължи да бъде изпълним. Това означава, че дори областта на

специализация на преводача да не включва информационни технологии, от него/нея все пак ще се изисква да притежава т.нар. дигитална компетентност.[1] Настоящият текст цели да разгледа реални, взети от практиката примери за съдържащи програмен код текстове с различни нива на сложност, като обясни начина на работа и даде полезни съвети за справянето с предизвикателството, което те могат да представляват както за бъдещия, така и за настоящия преводач.

Ще започнем с идентифицирането на непреводимите структури, което в някои случаи може да се окаже нелека задача, поради простия факт, че програмният код до голяма степен се състои от съвсем стандартни думи на английски език. Това означава, че в случаите на извършване на превод от английски на български някои непреводими кодови думи лесно биха могли да бъдат объркани за преводими такива, което в крайна сметка да доведе до неизпълним от компютъра кодов блок, т.е. до неизползваем превод. Ситуацията е до известна степен по-облекчена, когато преводът се извършва от или на езици, различни от английския, тъй като там би следвало да е по-трудно да бъдат объркани непреводимите кодови думи, но все пак не бива да забравяме, че програмистите, които пишат изпълнимия код, нерядко дават на променливите имена, съставени от пълнозначни думи на собствения си език.

Когато един текст се показва на компютърен екран, независимо дали на настолен, или мобилен компютър и независимо от използваната операционна система, този текст често бива визуализиран с помощта на браузър – програмен продукт, който чете и интерпретира изходен код, написан на езика HTML, който освен това може да съдържа код на езиците CSS и Javascript. Преди тази комбинация от HTML, CSS и Javascript да достигне до браузъра обаче, тя обикновено преминава през етап на обработка на сървъра, който изпраща този код към браузъра едва след като е изпълнил сървърния програмен код. Този сървърно-изпълним код може да бъде написан на редица езици – предимно на PHP, C, C++, Java, ASP.NET и прочие. Комбинации от човешки текст и компютърен код се използват, разбира се, и в много други случаи, като например приложения за настолни компютри, умни устройства и т.н., а наборът от използвани програмни езици се разширява още повече; ще разгледаме и такъв тип преводачески примери. Без значение кой програмен език е използван, преводачът трябва да умее да идентифицира непреводимите думи и/или структури и да борави с тях така, че след изпълнение на кода, автоматично вмъкнатият от тях текст да се съчетава с вече преведения такъв по граматически правилен начин в едно смислово цяло.

Една от основните цели на съчетаването на човешки език с компютърен код е персонализация на текста чрез включване в него на подробности за потре-

[1] Дигиталната компетентност може да бъде широко дефинирана като уверената, критична и креативна употреба на информационни и компютърни технологии за постигането на цели, свързани с работата, полезността, учението, развлечението, включването и/или участието в обществото (Ferrari 2013: 2).

бителя, който е адресат на този текст. В най-елементарния пример този текст ще се състои от едно изречение, съдържащо потребителско име, местоположение, адреса на компютъра (IP) и часа и датата на последното влизане на потребителя в системата.

Hello $username, your last login was at $time, $date from $location (IP: $IP).

В този пример освен думите на английски имаме и променливи на изпълнявания на сървъра език PHP, които сме способни да разпознаем благодарение на символа $, с който започва всяка от тях. Този символ е единственото, което отличава непреводимите променливи $username, $time, $date и $location от преводимите английски думи username, time, date и location. В българския си превод горното изречение ще изглежда по следния начин:

Здравей, $username. Последното ти влизане в системата беше в $time ч. на $date г. от $location (IP адрес: $IP).

Когато програмните променливи бъдат заменени с конкретните персонализирани данни и текстът бъде визуализиран в клиентския браузър, той ще изглежда по следния начин:

Здравей, Samurai. Последното ти влизане в системата беше в 20:36 ч. на 16.08.2015 г. от София (IP адрес: 85.130.104.15).

След като сме идентифицирали променливите, ние превеждаме останалата част от текста, прилагайки българските правописни правила, в това число поставянето на запетая след обръщението „Здравей“ и добавянето на съкращенията за час и година, които по стандарт изписваме след съответния час и дата; освен това сме направили и пояснителен превод на IP като „IP адрес“, макар това да не е стриктно необходимо. Виждаме също така и че за прегледност сме обособили съобщението в две самостоятелни изречения, въпреки че на английски имаме само едно.

Възможно е обаче създателите на конкретния сайт, в който сме влезли, да искат да предоставят на потребителя по-голяма функционалност, която да включва възможността потребителят да кликне върху потребителското си име, за да отиде на страница, съдържаща информация за неговия профил, както и възможността да кликне върху IP адреса и да види географска карта, на която този адрес е отбелязан с по-голяма точност. В този случай съобщението от по-горе би изглеждало по следния начин:

** Hello <a href=”index.php?page=profile&user=<?php echo $username; ?>”**

```
class="blue" title="Click here to view your profile info"><?php
echo $username; ?></a>, your last login was at <?php echo $time;
?>, <?php echo $date; ?>from <?php echo $location; ?>(IP: <a
href="#" title="Click here to see detailed information about this IP
address" onclick="Popup.show('<div>You will be taken to an
external website. Please click Understood to acknowledge or click
Cancel to prevent the new window from opening.<br />
><form><input type="button" onclick="location.href='http://
whatismyipaddress.com/ip/<?php echo $IP; ?>';"
value="Understood" /><input type="button" value="Cancel"
onClick="window.close()"></div>');return false;"
class="red"><?php echo $IP; ?></a>).</div>
```

В този текст вече, освен познатите ни от по-горе променливи в PHP, се съдържат и тагове от HTML и команди от JavaScript, а добавянето на друг код е наложило поставянето на тези променливи в PHP между отварящи и затварящи PHP тагове. Видимо, тук е необходимо да подходим с по-голямо внимание към идентифициране на преводимите елементи. В настоящия пример, освен съобщението, което вече преведохме по-горе, имаме и два атрибута на HTML тагове, чието съдържание следва да бъде преведено, и това са: *title* и *value*. Така преведеният на български текстови блок ще изглежда по следния начин:

```
<div class="container" id="welcome">Здравей, <a
href="index.php?page=profile&user=<?php echo $username; ?>"
class="blue" title="Кликни тук, за да видиш профилната си
информация"><?php echo $username; ?></a>. Последното ти
влизане в системата беше в<?php echo $time; ?> ч. на<?php
echo $date; ?> г. от<?php echo $location; ?> (IP адрес: <a
href="#" title="Кликни тук, за да видиш подробна информация
за този IP адрес" onclick="Popup.show('<div>Ще бъдеш
прехвърлен(а) към външен сайт. Моля, кликни \"Разбирам\", за
да потвърдиш или \"Отмени\", за да предотвратиш отварянето
на нов прозорец.<br /><form><input type="button"
onclick="location.href='http://whatismyipaddress.com/ip/<?php
echo $IP; ?>';" value="Разбирам" /><input type="button"
value="Отмени" onclick="window.close()"></div>');return false;"
class="red"><?php echo $IP; ?></a>).</div>
```

Атрибутът *title* на всеки HTML таг, който го поддържа, съдържа пояснителен текст, показван на потребителя, когато същият постави курсора на мишката си върху такъв елемент, но без да кликва върху него; текстът се скрива, когато курсорът излезе извън границите на този елемент. В нашия пример, ако

потребителят кликне върху връзката **85.130.104.15**, която представлява неговиятIP адрес, ще се отвори прозорец, пояснящ, че ако кликне върху бутона „Разбирам“, ще се отвори нов прозорец, в който да бъде показана подробна информация за този IP адрес, а ако кликне „Отмени“, този прозорец ще се затвори. Обърнете внимание, че правописните правила на български изискват всяко цитиране на външен текст, каквото представляват надписите на бутоните, да бъде поставено в кавички. Но тъй като кавичките – както единичните, така и двойните – се използват в програмните езици, за да обозначат стойност на атрибут или на променлива, директното поставяне на кавички около думите „Разбирам“ и „Отмени“ ще каже на интерпретатора на кода, че в тях се съдържа някаква важна за програмата стойност, а непредназначен за потребителя текст[2], като неправилното поставяне на кавички би могло да доведе до проблем с изпълнението на целия код, който следва по-надолу. Затова прибягваме до подход, наречен „избягване на кавичките“ (escape quotes) – също така познат сред програмистите като „искейпване“/„искейпвам“, който включва поставянето на обратна наклонена черта пред кавичката, която програмният интерпретатор не следва да чете.

Ето няколко примера за HTML тагове, които могат да съдържат атрибути, чието съдържание да изисква превод:

Таг	Значение	Примерно съдържание (опростено)
<a>	Превръща даден елемент, най-често текст, във връзка.	Начална страница
<img>	Показва изображение	<imgsrc="images/mona_lisa.jpg"title="Прочутата картина на Леонардо да Винчи" alt="Мона Лиза" />
<input>	Предоставя на потребителя възможност да въведе текст (потребителско име, парола, дата, телефонен номер и пр.) или да кликне бутон.	<input type="text" name="name"title="name" /> <input type="button" name="submit" value="Изпрати"title="Изпрати формуляра" />
<abbr>	Въвежда съкращение и неговото обяснение.	<abbr title="Европейска икономическа общност">ЕИО</abbr>
	Обособява секция с някакво съдържание.	JavaScript Kit

[2] Например атрибутът *type* и неговата стойност *button* указват на браузъра, че тагът *input* следва да бъде изобразен като бутон, а стойността *text* на същия този атрибут указва да бъде изобразено текстово поле, в което потребителят да може да въведе някаква информация.

<map>	Показва карта.	<map name="primary"title="Карта на Европа"> <areashape="rect" coords="200,250,25" href="detailed_map.html" /> <area shape="default" nohref /> </map>

Таблица 1: *Тагове с преводими атрибути*

Атрибутите *title* и *alt* се превеждат във всички случаи, в които съдържат чист текст. Трябва да отбележим обаче, че е възможно този текст да бъде генериран автоматично, например чрез извличане на информация от база данни посредством MySQL и PHP. В такъв случай стойността на атрибута ще представлява програмна променлива:

<imgsrc=”images/<?php echo $img; ?>” title=”<?php echo $img_desc; ?>” alt=”<?php echo $img_name; ?>” />

тоест в примера по-горе няма преводими елементи. Стойностите на атрибутите *title*и *alt* ще се зареждат от база данни, което ще рече, че ако клиентът се нуждае от техен превод, той трябва да осигури подходящ интерфейс за целта.

В предоставения за превод текст може да има най-различни непреводими елементи, било то HTML тагове или контейнери за автоматично генерирана информация (placeholders)[3] от друг програмен език:

**Your answers have been saved.

A confirmation email has been sent to the address you provided: [%%USER_EMAIL%%]**

Тук имаме два HTML тага за преминаване на нов ред (
 или
) и един контейнер, на чието място програмният код ще постави имейл адреса на потребителя. Извършвайки превода, поставяме двата тага за нов ред след точката на първото изречение. Във второто изречение обаче е необходима промяна, която е обусловена от българския словоред. В английския текст имаме “confirmation email”, което се превежда като „имейл за потвърждение“, а самият адрес се отпечатва логически след “… you provided:” Ако не направим разместване на контейнера, резултатът ще бъде следният:

[3] Както е типично за компютърните термини, те често се транскрибират, вместо да се търси техен превод – практика, която категорично не поощряваме. В случая е възможно, най-вече сред самите програмисти, да се чуе „плейсхолдър“ вместо „контейнер“.

**Вашите отговори са запазени.

На посочения от Вас адрес е изпратен имейл за потвърждение: [%%USER_EMAIL%%]**

Така обаче имейл адресът на потребителя ще бъде показан на смислово неудачно място и затова се налага разместване на контейнера и ограждането му с тирета за обособяване:

**Вашите отговори са запазени.

На посочения от Вас адрес – [%%USER_EMAIL%%] – е изпратен имейл за потвърждение.**

В примера по-долу контейнерът [xx] ще бъде заместен със сума, която може да бъде спечелена от участващия в даден турнир играч:

Also, remember that our awesome [xx] weekly prize contests every Sunday are EXCLUSIVE for gold members – so for [xx] you could take part in this contest and make some money every week.

Докато във втория случай не е необходимо никакво разместване на позицията на контейнера – „така че срещу [xx] ще можеш да участваш в това състезание и да печелиш пари всяка седмица“ – то в първия това не е така, защото в „[xx] weekly prize contests“ контейнерът [xx] изпълнява граматическата функция на определение, което налага следния превод:

Не забравяй също така, че страхотните ни ежеседмични състезания с награда от [xx] всяка неделя са ЕКСКЛУЗИВНИ за „златни“ членове…

а не:

Не забравяй също така, че страхотните ни [xx]ежеседмични състезания с награда всяка неделя са ЕКСКЛУЗИВНИ за „златни“ членове…

Когато контейнерът бъде заместен с реална стойност, ще получим:

Не забравяй също така, че страхотните ни ежеседмични състезания с награда от 50 долара всяка неделя са ЕКСКЛУЗИВНИ за „златни“ членове…

Контейнерите или променливите могат да бъдат обозначавани по най-различни начини: оградени в една или две къдрави скоби, в квадратни скоби, със знаците $, % или двоеточие пред думата и т.н., вж. таблицата с примери по-долу:

Изходен текст	***Преводен текст***
{0} out of {1} tracks processed	Обработени са {0} от {1} парчета
Group {group} isnotok!	Групата {group} не е ОК!
Installing {{appname}}...	{{appname}} се инсталира...
The field is [c123] characters too long	Полето е с [c123] по-дълго от необходимото
Who'sgoing to be at [Bar Name] tomorrow night for the first of our Famous Destination Parties.	Кой ще бъде утре вечер в [Bar Name] на първото от нашите Famous Destination партита?
<string name="wifi_list_title_one">%d open WiFi found</string>	<string name="wifi_list_title_one">Намерена е %d отворена безжична мрежа</string>
:appName :couponValue Coupon For Free!	Безплатен ваучер за :couponValue за :appName!
We do not see or store your photos.	Ние не виждаме и не съхраняваме Вашите снимки.

Таблица 2: *Примери за текст, съдържащ контейнери*

В последния пример е използван HTML тагът *span*, на който е зададен атрибутът *class* със стойност *underline*, което означава, че върху думата, намираща се между отварящия и затварящия таг, ще бъде приложен CSS стил за подчертаване. Тъй като българският превод налага повтарянето на отрицателната частица „не“ два пъти, за разлика от еднократното“not” в английския текст, преводачът трябва да приложи същото форматиране и върху втората частица „не“, като за целта копира целия HTML таг и го постави, където е необходимо.

Тагът за преминаване на нов ред в HTML е, както казахме,
 или
, но е възможно това преминаване на нов ред да е заложено още в изпълнявания на сървъраPHP код, т.е. преди текстът да стигне до браузъра. В PHP за тази цел се използват специалните знаци **n** или **r**, които, за да не бъдат разбрани буквално като буквите **n** и **r**, се „искейпват“с обратна наклонена черта (**\n** и **\r**) и често се употребяват заедно в следната последователност: **\r\n**. Редовно се срещат в даден ни за превод текст:

WARNING!\n\nThis “ + (getURLParameter(“brand”)) + “ “ + (getURLParameter(“model”)) + “ is infected with virus and serious damage is done to your battery.
ПРЕДУПРЕЖДЕНИЕ!\n\nТози “ + (getURLParameter(“brand”)) + “ “ + (getURLParameter(“model”)) + “ е заразен с вирус и батерията ти е сериозно увредена.

Виждаме, че тук знакът за нов ред **\n** е употребен два пъти, т.е. между „ПРЕДУПРЕЖДЕНИЕ“ и следващия текст ще бъдат оставени два празни реда. Обръщаме внимание и на променливите (параметрите) *brand* и *model*, които се използват от функцията *getURLParameter()* и които, ако преводачът не идентифицира като непреводими и преведе като „марка“ и „модел“, ще доведат до неизпълнение на кода, т.е. до неизползваем превод.

Вече споменахме, че единичните и/или двойните кавички се употребяват в програмния код за въвеждане на стойности на променливи, като неправилното им поставяне може да причини проблеми с изпълнението на кода. Същото се отнася и за знака апостроф (‘)в английския, който се използва както за обозначаване на принадлежност, така и за съкращаване на спомагателен глагол. В примесен с код текст е необходимо избягването този апостроф чрез поставянето на обратна наклонена черта пред него, за да не бъде разбран от интерпретатора като единична кавичка:

$correct_answer = ‘Congratulations!\r\nThat\’s the right answer!’;
$wrong_answer = ‘Sorry!\r\nThat\’s wrong, try again!’;

В превода на български обаче този апостроф не е необходим, затова го изпускаме, а заедно с него и наклонената черта, служеща за избягването му:

$correct_answer = ‘Поздравления,\r\nтова е правилният отговор!’;
$wrong_answer = ‘Съжаляваме,\r\nотговорът е грешен. Опитайте отново!’;

В горния пример обаче запазваме специалните знаци за нов ред **\r\n**. Същото се отнася и за случаи, в които апострофът е използван, за да обозначи притежание – в преводния текст той не ни е нужен:

$user has sent you a message. Click $link to read $user\’s message.
$user ти е изпратил съобщение. Кликни $link, за да прочетеш съобщението на $user.

Възможно е да се озовете в ситуация, в която клиентът да е подготвил текст за превод, който не взема предвид особеностите на целевия език. В такъв случай единственият изход е сами да потърсите решение на проблема. Нека разгледаме следния пример:

‘s best friend is

Това изречение е част от по-голям текст, в който потребителите на Facebook биват подканени да използват развлекателно приложение, което да им покаже кой е най-добрият им приятел. Възложителят на този превод изрично е указал, че „в някои от низовете[4] ще бъде автоматично добавено име/имена преди или след низа или както преди, така и след, но никога вътре в низа“. Когато към горния низ бъдат добавени имената, той ще изглежда по следния начин:

George‘s best friend is David

Бихме го превели по следния начин:

Най-добрият приятел на George е David

Програмата на клиента обаче работи по такъв начин, че името на потребителя се добавя в началото, а името на най-добрия му приятел – в края на низа, което, ако използваме горния превод, ще доведе до следното:

George Най-добрият приятел на е David

което несъмнено е неадекватно. Единственото възможно в случая решение е да изменим превода леко и го перифразираме като обръщение:

, твоят най-добър приятел е

Заедно с автоматично добавените имена, низът ще изглежда по следния начин:

George, твоят най-добър приятел е David

което вече представлява граматически правилно и смислено изречение.

Понякога обаче е невъзможно преводачът да намери решение. В такъв случай е редно той/тя да уведоми клиента за проблема и да предложи по-

[4] Низ – от английски string (text string) – е поредица от символи и знаци, която интерпретаторът на програмния код приема за не-код, т.е. текст, който не се изпълнява, а само се отпечатва на екрана.

нататъшно съдействие, ако е необходимо. Нека разгледаме следното на пръв поглед доста лесно задание за превод, в което е необходимо да се направи превод на съобщение, показвано при изтегляне на файл, което на английски би изглеждало така:

Downloading: 12/142MB, 14 minutes left.

А българският му превод така:

Изтегляне: 12/142 MB, остават 14 минути.

Клиентът е предвидил за превод следните думи и низ, като ги е изпратил във файл от програмата Excel; показваме ги заедно с превода им:

String	**Bulgarian Translation**
second	секунда
seconds	секунди
minute	минута
minutes	минути
hour	час
hours	часа
day	ден
days	дни
kB	kB
MB	MB
GB	GB
{time} left	остават **{time}**

Таблица 3: *Клиентски текст и неговият превод*

Тук обаче имаме проблем и той се намира в низа „остават **{time}**“, тъй като той не покрива варианта, в който оставащото време ще започва с „1 ден“, „1 час“ или „1 минута“. Граматически правилното изречение на български в такъв случай би било:

Изтегляне: 12/142 MB, остава 1 час и 15 минути.

Тоест, необходимо е клиентският код да може да прецени кога оставащото време ще започва с цифра „1“ и тогава да използва формата на глагола „остава“, а не „остават“. В конкретната ситуация единственото, което можеше да се направи (и което аз направих), е да се обясни това на клиента и да се опишат вариантите:

{time} left	остават **{time}**	PLURAL
	остава **{time}**	SINGULAR: 1 day, X hour(s), X second(s)
		SINGULAR: 1 hour, X minute(s), X second(s)
		SINGULAR: 1 minute, X second(s)
		SINGULAR: 1 second

Таблица 4: *Пояснения към превод на низ, съдържащ променлива*

По-горе са изброени всички възможни комбинации, които изискват формата „остава“. Оттук нататък, нещата са изцяло в ръцете на клиента. Освен горното описание обаче аз добавих и коментар, че ако е непрактична промяната на кода по такъв начин, че да вземе предвид тази особеност на българския превод, а може би и на други езици, то тогава може да бъде използвана само формата „остават“, въпреки грешката, защото тази грешка не е съществена, т.е. не променя смисъла на съобщението.

Нерядко текстът, който трябва да се преведе, се предоставя в XML файл, в който преводимото съдържание е ясно отделено от непреводимото. Например тук:

```
<string name=”guide_failed_tip_text”>Please upgrade to the latest
version to\napply this theme</string>
<string name=”guide_failed_update_button_text”>Update</string>
<string name=”lbl_share”>Share</string>
<string name=”share_skin_download_text1">Check my #Flash
Keyboard</string>
<string name=”share_skin_download_text2">Isn\’t that amazing?</
string>
```

Това, разбира се, прави задачата много по-лесна: превеждаме това, което се намира помежду отварящия и затварящия таг <string>. Имената на таговете, посочени в параметъра **name=””** са също непреводими единици, което е видно от използваните долни черти за свързване на думите в едно цяло наименование, напр. **guide_failed_tip_text**. В този пример трябва да внимаваме с две неща: 1. **#Flash Keyboard**, което е името на приложението и като такова не се превежда, още повече че пред него е поставен знакът диез (хаштаг), с който

се обозначават теми и/или канали във Фейсбук или Туитър; и 2. апострофът за съкращаване на глагола, който е избегнат с обратна наклонена черта в **Isn\'t that amazing?** Огромно улеснение за преводача представляват преводаческите инструменти (CAT tools или Computer Assisted Translation tools, инструменти за превод с помощта на компютър[5]), които умеят да различават преводимите от непреводимите единици в някои видове файлове, напр. XML, HTML, YML и пр., но за съжаление не могат да се справят с код, който е вграден директно в текста, както във всички разгледани по-горе примери.

Сходен пример откриваме и в следния блок:

```
"refer": {
"title": "Share this opportunity",
          "linkedin": {
             "subject": "Could this be of interest?",
            "body": "Hi {firstName},\n{company} is hiring a {job}. I
thought you might find this interesting. You can view the job ad here: {link}"
          }
}
```

Този блок е извадка от по-голям JSON файл, който представлява формат за обмен на данни за езика Javascript и който се поддържа – донякъде – от преводаческите инструменти. Под „донякъде“ имам предвид, че удебеленият текст ще бъде разпознат като непреводим и ще бъде изцяло скрит от преводача, като ще бъде показан само текстът, който е поставен между кавичките. Виждаме обаче, че в него също така са включени променливи, които не се превеждат – в елемента **"body"**:

Hi {firstName},\n{company} is hiring a {job}. I thought you might find this interesting. You can view the job ad here: {link}

Тук firstName, company, job и link представляват променливи или контейнери (placeholders), които трябва да бъдат оставени в превода на подходящото, логическо, място – за щастие в случая то не се променя:

Здравей, {firstName},\n{company} търси да наеме {job}. Реших, че това може да ти представлява интерес. Можеш да видиш обявата за позицията тук: {link}

Ако обаче преводачът, поради една или друга причина, не разполага с преводачески софтуер, той/тя ще трябва да работи директно в JSON файла, който

[5] Разглеждам работата с тях в статията си „Преводаческите инструменти в образованието“ (Стефанов 2012).

ще трябва да отвори с някой от тъй наречените „богати текстови редактори“, например EditPlus или Notepad++, които отделят визуално (като ги оцветяват различно)класовете програмен код и текстови (преводими) низове. Макар да са предназначени за напреднали програмисти, тези богати текстови редактори се използват – по необходимост – и от преводачите.

На екранната снимка (скрийншота) по-долу е показана част от файл, написан на програмния език PHP, който съдържа няколко преводими низа:

*** Project has been successfully written.**
*** Project write failed:**
*** Payable successfully registered.**
*** ERROR: Payable registration failed**
*** Project has been updated successfully.**

```
        }
        include("project_form.php");
    }
    // if no errors, write project data to database
    else {
        if($_POST['do'] == "New") {
            if($db->query("INSERT INTO ?n SET ?u", PROJECT_TABLE, $project_data)) {
                $system_feedback = "* Project has been successfully written. ";
            }
            else {
                $system_feedback = "* Project write failed: ".$db->lastQuery();
            }
            // register payable
            if($project_data['vendorID'] != "1") {
                $projectID = $db->insertId();
                $payable_data = array(
                    'projectID' => $projectID,
                    'date'      => $project_data['duedate'],
                    'vendorID'  => $project_data['vendorID'],
                    'subprice'  => $project_data['subprice'],
                    'totaltopay'=> ($project_data['subprice'] * $project_data['units']),
                    'currency'  => $project_data['currency'],
                    'paid'      => '0'
                );
                if($db->query("INSERT INTO ?n SET ?u", PROJECT_PAYABLES, $payable_data)) {
                    $system_feedback .= "* Payable successfully registered.";
                }
                else {
                    $system_message .= "* ERROR: Payable registration failed";
                }
            }
            include("project_list.php");
        } // end new project
        elseif($_POST['do'] == "Edit") {

            if($db->query("UPDATE ?n SET ?u WHERE projectID = ?i", PROJECT_TABLE, $project_data, $project_data['projectID'])) {
                $system_feedback = "* Project has been updated successfully. ";
            }
```

Фигура 1: *Извадка от програмен код, съдържащ преводим текст*

Всичко останало е програмен код, който преводачът не бива да променя по никакъв начин.

В някои случаи, с оглед именно улесняване работата на преводача, всички текстове, показвани в даден софтуер, се извеждат в самостоятелни файлове. Тези файлове могат да имат различни разширения, напр. .ini (за различни програмни езици), .xlf (за различни инструменти), .properties (за Java), .resx (за .NET), вече споменатия .yml (за Ruby on Rails), .strings (за iOS и MacOS X) и др.[6] Структурата

[6] За подробен списък, вж. https://www.getlocalization.com/docs/file-formats/

на тези файлове е сравнително праволинейна: от лявата страна е името на променливата, а нейната стойност за съответния човешки език се присвоява чрез знак за равенство, двоеточие и др., като в някои случаи самата променлива и нейната присвоена стойност могат да бъдат поставени в кавички. По-долу виждаме извадка от .strings файл:

```
“RelativeDateDativeYears_1” = “1 year”;
“RelativeDateDativeYears” = “%ld years”;
“RelativeDateDativeMonths_1” = “1 month”;
“RelativeDateDativeMonths” = “%ld months”;
“ReceivedBadgesTitle” = “CONGRATULATIONS!”;
“ReceivedBadgeMessage” = “You have received the “%@” badge.”;
“ReceivedBadgesMessage” = “You have received the “%1$@” badge and %2$d others.”;
```

И преводът на съдържанието на извадката на български:

```
“RelativeDateDativeYears_1” = “1 година”;
“RelativeDateDativeYears” = “%ld години”;
“RelativeDateDativeMonths_1” = “1 месец”;
“RelativeDateDativeMonths” = “%ld месеца”;
“ReceivedBadgesTitle” = “ПОЗДРАВЛЕНИЯ!”;
“ReceivedBadgeMessage” = “Ти получи значката “%@”.”;
“ReceivedBadgesMessage” = “Ти получи значката “%1$@” и още %2$d други.”;
```

и от .ini файл:

```
COM_MEDIA_UPLOAD=“Upload”
COM_MEDIA_UPLOAD_COMPLETE=“Upload Complete”
COM_MEDIA_UPLOAD_FILE=”Upload file”
COM_MEDIA_UPLOAD_FILES=”Upload files (Maximum Size: %s MB)”
COM_MEDIA_UPLOAD_FILES_NOLIMIT=“Upload files (No maximum size)”
COM_MEDIA_UPLOAD_SUCCESSFUL=“Upload Successful”
```

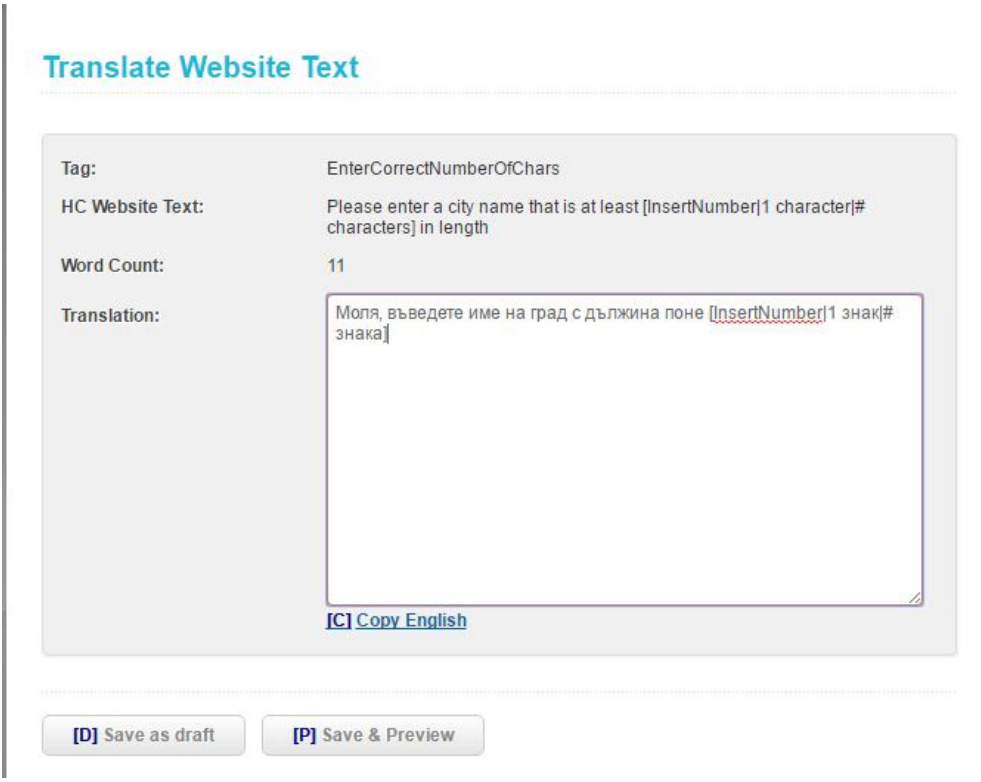
и превод на съдържанието на извадката на български:

```
COM_MEDIA_UPLOAD=“Качи”
COM_MEDIA_UPLOAD_COMPLETE=“Качването завърши”
COM_MEDIA_UPLOAD_FILE=“Качи файл”
```

COM_MEDIA_UPLOAD_FILES=“Качи файлове (максимален размер: %s MB)”
COM_MEDIA_UPLOAD_FILES_NOLIMIT=“Качи файлове (без ограничение на макс. размер)”
COM_MEDIA_UPLOAD_SUCCESSFUL=“Качването е успешно”

Както виждаме, дори в тези специални файлове не е изключено в преводимата част да се съдържат елементи от програмния код, които не бива да се превеждат, а следва да бъдат поставени на точното им място според правилата на съответния език, на който превеждаме.

За по-нататъшно улеснение както на преводача, така и на възложителя и неговите разработчици, всички преводими текстове могат да бъдат поставени на отдалечен сървър, в който преводачът влиза със собствено потребителско име и парола и превежда само възложените му/ѝ за дадена дата например елементи. По-долу е показана снимка на екрана от една такава платформа за локализация:



Фигура 2: *Екранна снимка на платформа за локализация*

Платформата показва името на контейнера (тага), която в този случай е наименувана EnterCorrectNumberOfChars, нейното съдържание на английски и преводът на това съдържание на български. Виждаме също така, че конкретният таг съдържа променлива, която приема различна стойност спрямо броя знаци, т.е. показва една форма за единствено число и друга – за множествено.

Изводът от всичко казано дотук е, че за да бъде адекватен на изискванията на пазара в началото на 21-ви век, преводачът не просто трябва да може да работи с компютър, а да притежава и едно приемливо ниво на компетентност в областта на информационните технологии, по-конкретно в езиковото обезпечаване на софтуера и уебсайтовете. Да умее да различава преводни от непреводни елементи, т.е. човешки от програмни езици, и да разбира начините на съчетаването им за персонализиране на съобщенията спрямо индивидуалния потребител или спрямо конкретната ситуация. За да се усвоят тези умения е необходимо адаптирането на учебните програми в университетите към новото време – включването на специализирани курсове по превод на текст, съдържащ код – и евентуално предлагането на допълнителни курсове за разширяване квалификациите на вече завършилите, което от своя страна би осигурило допълнителни приходи за съответната образователна институция и би ѝ дало възможност да се включи в програмите за учене през целия живот, което в контекста на съвременното общество на знанието е важно не само от гледна точка на „конкурентоспособността и пригодността за заетост, но също така и за социално включване, активно гражданство и личностно развитие“ (Commission of the European Communities 2006).

ЛИТЕРАТУРА // BIBLIOGRAPHY

Стефанов 2012: Стефан Стефанов, „Преводаческите инструменти в образованието“, в сборник от *Международна научна конференция по теория и практика на превода в памет на гл. ас. Даниела Димитрова Петрова*, 24-25 ноември 2011 г., ВТУ „Св. св. Кирил и Методий“, В. Търново: Фабер, 2012, с. 277-287, ISBN 978-954-400-671-9. Stefanov, Stefan. “Prevodacheskite instrumenti v obrazovanieto” in the journal of proceedings of *International Scientific Conference in Translation Theory and Practice in Memoriam of Senior Lecturer Daniela Dimitrova Petrova*, 24-25 November 2011, St Cyril and St Methodius University of Veliko Tarnovo, Veliko Tarnovo: Faber Publishing, 2012, pp. 277-287, ISBN 978-954-400-671-9.

Commission of the European Communities. *Communication from the Commission – Adult learning: It is never too late to learn.* EUR-Lex: 2006. 13.10.2017. <http://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:52006DC0614>.

Ferrari, Anusca. *DIGCOMP: A Framework for Developing and Understanding Digital Competence in Europe.* Ed. Punie, Yves and Brenko, Barbara N. JRC Scientific and Policy reports. European Commission: 2013. 13.10.2017 г. <http://ftp.jrc.es/EURdoc/JRC83167.pdf>.