Розглядатимемо тепер інтервал (-1,1). Очевидно, на ньому (як і на будь-якому іншому) функції 1, x, x^2, x^3…утворюють систему Маркова. Можна показати, що в такому випадку $\boldsymbol{Q_n(x)=\frac{\sin(n+1)\varphi}{\sin\varphi}, x_r=\cos\varphi, r=1,2..n}$.

Відшукаємо многочлен найкращого наближення 1-го степеня для функції $f(x)=e^x$, k=1 x_1

$$\begin{cases} \boldsymbol{Q_2=\frac{\sin 3\varphi}{\sin\varphi}} \\ \boldsymbol{x_1=\cos\varphi} \end{cases}$$

Виконавши елементарні перетворення, отримаємо,

$$\boldsymbol{Q_2(x)=3-4\sin^2\varphi=3-(4-4\cos^2\varphi)=3-4+4x_1=4x_1-1}$$
$$\boldsymbol{Q_2(x_1)=0}$$
$$\boldsymbol{x_1=\frac{1}{4}.}$$

Знайдемо мінімальне відхилення з теореми Маркова. Будемо мати,

$$\left|\int \boldsymbol{f(x)signQ_2(x)dx}\right| = \left|\int_{-1}^{1/4} -\boldsymbol{e^x dx} + \int_{1/4}^{1} \boldsymbol{e^x dx}\right| = \tag{1}$$
$$= \left|-(\boldsymbol{e^{1/4}-e^{-1}})+(\boldsymbol{e-e^{1/4}})\right| = \left|\boldsymbol{e-2e^{1/4}+e^{-1}}\right| = \boldsymbol{e-2e^{1/4}+e^{-1}},$$

тобто маємо відхилення порядку 0,5. З теореми випливає, що графіки $y=e^x$ і $y=\alpha+\beta x$ перетинаються в $x_1=\frac{1}{4}$ (*). Зважаючи на те, що $y=e^x$ швидко зростає можемо стверджувати, що, по-перше, $\beta>0$, по друге, лівіше від точки $x_1=\frac{1}{4}$ графік $y=\alpha+\beta x$ знаходиться вище за графік $y=e^x$, а правіше – навпаки. Тому відхилення від $y=\alpha+\beta x$ від $y=e^x$ будемо шукати у вигляді

$$\int_{-1}^{1/4}(\alpha+\beta x-e^x)dx+\int_{1/4}^{1}(e^x-\alpha-\beta x)dx = 2\alpha\frac{1}{4}+\beta(\frac{1}{16}-1)+e+e^{-1}-2e^{1/4} \tag{2}$$

Порівнюючи (1) і (2), матимемо, що $\frac{\alpha}{2}-\frac{\beta}{16}=0,$ або $\beta=8\alpha$. З (*) робимо висновок, що

$\alpha+\frac{\beta}{4}=e^{1/4},$ звідки $\alpha=\frac{e^{1/4}}{3}, \beta=\frac{8}{3}e^{1/4}.$

Отже, на [a,b] функцію $y=e^x$ з усіх алгебраїчних многочленів першого степеня найкраще наближає $\boldsymbol{y=\frac{e^{1/4}}{3}+\frac{8}{3}e^{1/4}x}$.

Отже, пошук многочлена найкращого наближення (нагадаємо, що тут маються на увазі не суто алгебраїчні многочлени) носить певні труднощі. В окремих випадках для відшукання такого многочлена можемо застосовувати теорему Маркова, проводячи оцінку мінімального відхилення та шукаючи многочлен, для якого така оцінка справедлива.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ахиезер Н. И. Лекции по теории аппроксимации. – Москва: Наука, 1965. – 408 с.
2. Дзядык В. К. Введение в теорию равномерного приближения функций полиномами – Москва: Наука, 1977. – 512 с.

Панчук О.

Науковий керівник – доц. Олексюк В. П.

ОСОБЛИВОСТІ ВИКОРИСТАННЯВЕБ-ФРЕЙМВОРКАСАКЕРНР ПРИ СТВОРЕННІ API-ФУНКЦІЙ ДЛЯ РОБОТИ З МОБІЛЬНИМИ ПРИСТРОЯМИ

Сьогодні роль мобільного телефону в житті людини важко переоцінити. За допомогою цього компактного пристрою ми можемо не лише швидко зв'язатися зі своїми родичами, друзями, чи колегами по роботі, а й отримати доступ до мережі Інтернет. Проте робота в Інтернеті з мобільного пристрою має свої особливості порівняно з доступом з традиційного комп'ютера: даються взнаки і швидкість мобільного Інтернету, і розширення екрану, і особливості опрацювання вказівок. Тому виникає необхідність оптимізації веб-сайтів і веб-додатків для мобільних платформ.

Метою цієї статті є аналіз основнихпідходів організаціїінтерфейсівприкладного програмування (API —applicationprogramminginterface) призначених для роботи з мобільними платформами, та вивчення підходу REST (RepresentationalStateTransfer, передавання репрезентативних станів),як ефективного для використання передавання даних з веб-сервера на телефон. Також розглянемо php-фреймворкCakePHP, як основу для створення REST API.

Актуальність роботи полягає у вивченні ефективного підходу передавання даних із веб-серверів на мобільні телефони, а також в особливостях реалізації REST-функції за допомогою фреймворкуCakePHP.

Упроцесі роботи були використані такі **методи дослідження**: експеримент, спостереження, моделювання, висунення гіпотез.

Вивчаючи проблему розробки веб-сервісу, до якогозверталися б звичайні клієнти та інші веб-служби, варто відповісти на питання: як найкраще організувати API? Сучасні технології пропонують цілий ряд варіантів вирішення проблеми, різних за своєю структурою та ідеологією.Серед них слід виділити підходи REST, SOAP, XML-RPC та інші.

XML-RPC (ExtensibleMarkupLanguageRemoteProcedureCall, виклик віддалених процедур розширеною мовою розмітки) — стандарт/протокол виклику віддалених процедур, якийвикористовує розширену мову розмітки (XML) для кодування своїх повідомлень і протокол передачі гіпертексту(HTTP)як транспортний механізм. Інтерфейс XML-RPC єпрабатьком SOAP, відрізняється винятковою простотою застосування. XML-RPC, як і будь-який інший інтерфейс RPC, визначає набір стандартних типів даних і команд, які програміст може використовувати для доступу до функціональності іншої програми, яка виконується на іншому комп'ютері у мережі.

SOAP (SimpleObject Access Protocol, простий протокол доступу до об'єктів) — протокол обміну структурованими повідомленнями в розподіленому обчислювальному середовищі. Спочатку SOAP призначався переважнодля реалізації віддаленого виклику процедур (RPC). Зараз протокол використовується для обміну довільними повідомленнями у форматі XML, а не тільки для виклику процедур. SOAP є розширенням протоколу XML-RPC.

SOAP може використовуватися з будь-яким протоколом прикладного рівня: SMTP, FTP, HTTP, HTTPS, однак його взаємодія з кожним з цих протоколів має свої особливості, які повинні бути визначені окремо. Найчастіше SOAP використовується разом з HTTP.

SOAP є одним із стандартів, на яких базуються технології веб-служб.

Проаналізувавши сучасні статті та технології[1],[3],[4] можна зробити висновок, щоRESTдля вище поставленої проблеміє найзручнішою технологією, оскількиекономить обсяги даних,якіпередаються, а також за своєю архітектурою є більш «строгою», ніж інші технології, що полегшує написання структурно правильних API.

REST (RepresentationalStateTransfer, «передавання репрезентативного стану») — стиль побудови архітектури розподіленого додатку. Був описаний і популяризований у 2000 році Роєм Філдінгом (RoyFielding), одним з творців протоколу HTTP. Найвідомішою системою, яка побудованазначною мірою з архітектури REST, є сучасна Всесвітня павутина.

Згідно RESTдані в слід передавати у вигляді невеликої кількості стандартних форматів (наприклад HTML, XML, JSON). Мережевий протокол (як і HTTP) повинен підтримувати кешування, а також не повинен залежати від мережевого шару та зберігати дані про стан кожної пари «запит-відповідь». Такий підхід забезпечує масштабованість системи і дозволяє їй еволюціонувати з новими вимогами.

REST API визначає набір функцій, до яких розробники можуть здійснювати запити і отримувати відповіді. Взаємодія відбувається за протоколом HTTP. Перевагою такого підходу є широке поширення протоколу HTTP, тому REST API можна використовувати практично з будь-якої мови програмування.

Отож, відповіді від сервера REST можуть бути у форматі XML або JSON[4]. Пам'ятаючи про потребу мінімізувати обсяг інформації, що передається,доцільніше буде використовувати JSON, оскільки він більш лаконічний, ніж XML. Різниця може бути порядку декількох сотень кілобайт, але, з урахуванням швидкостей 3G і нестабільності обміну з мобільними пристроями, ці кілька сотень кілобайт можуть мати значення.

Всі запити до REST API формуються у окремо сформованому URL і мають вигляд:

http://yourdomain.com / api / <тип запиту> / <ім'я функції>? param1 = value& param2 = value ...

<тип запиту> може бути: xml або json або jsonp

Отже, визначившись, у якому форматі нам потрібно отримувати дані з сервера, виберемо метод написання API.

Технології стрімко поширюються, і з подібними завданнями стикаються програмісти зі всього світу. Реалізація всього коду «з нуля» часто є недоцільною і непрактичною, особливо з врахуванням значної кількості робочих помилок. Для полегшення написання веб-додатків існує ряд фреймворків.

Веб-фреймворки — це групи готових веб-компонентів і моделей, які полегшують веб-програмування і роблять його більш організованим. Використовуючи веб-фреймворк, можна значно спростити роботу, необхідну для побудови веб-додатків.

Усі веб-фреймворки використовують патерн проектування MVC (model-view-controller). Патерн проектування — це організація коду або компонентів, яка вирішує деяку проблему проектування, що виникла в робочому рішенні. Шаблон MVC є загальним у програмуванні веб-додатків і розділяє додаток на три частини. Перша частина являє собою бізнес-процеси (модель). Друга «говорить» додаткам, що робити з потоками даних (контролер). А за допомогою третьої частини створюються HTML-сторінки (вигляд). Багато веб-фреймворків, які використовуютьшаблон MVC, дозволяють організувати код так, щоб зміни у моделі, вигляді або контролері мали незначний вплив на інші елементи програми.

Прикладамивеб-фреймворків, доступних для мови програмування PHP, єCakePHP, CodeIgniter, Symfony і Zend Framework.

ФреймворкCakePHP може бути надійною основою для веб додатку. Він може керувати кожним аспектом, від першого запиту користувача до відображення даних на сторінці.

Фреймворк також забезпечує основну організаційну структуру, від імен файлу до імен таблиць бази даних, зберігаючи додаток послідовним і логічним. Але це, з іншого боку, є й недоліком CakePHP.Фреймворквимагає чіткого дотримання назв та угод. Він очікує певного іменування файлів, класів, баз даних, методів.

Проте у всьому іншому використовувати цей фреймворк дуже зручно.УCakePHP вбудовано цілий ряд вже готових реалізованих функцій[2]. Завдяки ООП при створенні нового додатку потрібно створити клас, який є нащадком вже створеного класу CakePHP.Після цього розробник може використовувати всідоступні методи цього класу.

ОскількиCakePHP реалізує модель MVC, для написання REST API потрібно буде створити контролер, модель, вигляд. Проте, як вже було згадано, REST-функція нам потрібна для отримання даних в форматі JSON, тому вигляду як такого нам не потрібно. Весь функціонал має знаходитися в контролері, клас якого буде нащадком класуAppController.

Наприклад, оголосивши

- *classNameControllerextendsAppController*
- користувач отримає всі доступні методи класу AppController.Далі можна застосувати ще одну дуже корисну особливістьCakePHP—автоматичнегенерування даних у форматі JSON.

Для прикладу нехай змінна$data містить масив даних, які необхідно передати у форматі JSON. У фреймворкуCakePHP скористаємось таким рядком.

- *$this->set('_serialize', 'data')*

Тобто, передавши у функцію через URL певні параметри і при умові, що ця функція повертає $data, ми отримаємо дані у форматі JSON.

Оскільки останнім часом великого значення набула персоналізація контенту, доцільно розглянути механізм автентифікаціїз мобільного пристрою за допомогою REST API, який реалізованозасобамиCakePHP. Зважаючи на той факт, що зараз більшість інтернет-ресурсів включають в себе велику кількість веб-сервісів пов'язаних між собою,розглянемо той випадок коли автентифікація реалізується з використанням LDAP-сервера.Користувачу, якийпройшовпроцедуру автентифікації, потрібно вивести певні персональні дані з бази даних. Для реалізації поставленого завдання нам необхідно отримати дані і відправити назад результат їх обробки на мобільний пристрій. У результаті опрацювання матеріалів та проведених досліджень, використовуючи фреймворкCakePHP,вдалося реалізувати REST API функцію, що й виконує поставлене завдання.

Існує цілий ряд проблем які вирішуються за допомогою фреймворкуCakePHP. Серед них не лише написання API для роботи з мобільними технологіями, а й створення найрізноманітніших веб-додатківта повноцінних сайтів. Використання CakePHP полегшує та спрощує роботу веб-програміста. Також швидкість обробки скриптів написаних на CakePHP є більшою ніж скриптів найпоширеніших CMS (Joomla, WordPress)

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. CodyFauser, JamesMacAulay. Rails 3 in a Nutshell. [Електронний ресурс]. – Режим доступу: http://ofps.oreilly.com/titles/9780596521424/activeresource_id59243.html
2. JSON and XML views. CakePHPbook.[Електронний ресурс].– Режим доступу:http://book.cakephp.org/2.0/en/views/json-and-xml-views.html
3. Архитектура REST.[Електронний ресурс].– Режим доступу:http://habrahabr.ru/post/38730/
4. Немного о том какорганизовывать API веб-службы.[Електронний жернал].– Режим доступу:http://habrahabr.ru/post/127243/

Островська Л. І.

Науковий керівник – доц. Кравчук В.Р.

НАБЛИЖЕННЯ СТЕПЕНЕВОЇ ФУНКЦІЇ МНОГОЧЛЕНАМИ ЗА *A*-МЕТОДОМ

Майже в усіх галузях прикладної математики важливу роль відіграють задачі апроксимації «більш складних» функцій «менш складними». Такі задачі виникають при обробці експериментальних даних, чисельному диференціюванні та інтегруванні тощо. Тому важливо знати основні методи, результати і задачі теорії наближення функцій.

Найбільш простим і найбільш вивченим апаратом наближення функцій є множина многочленів або (у випадку періодичних функцій) множина тригонометричних поліномів заданого порядку n. У використанні на практиці дуже зручними є многочлени. Щоб утворити многочлен досить задати тільки скінченну кількість його коефіцієнтів. Значення многочлена просто обчислити, його легко продиференціювати, проінтегрувати.

Якщо здійснюють наближення деякої функції многочленами і при цьому виходять лише із властивостей даної функції, то кажуть, що здійснюють апроксимацію даної функції. Існують різні методи апроксимації функцій многочленами, кожний з яких забезпечує певну «точність» наближення.

Наближення функцій можна здійснювати за допомогою частинних сум n-го порядку ряду Тейлора цих функцій, які отримуються з ряду Тейлора, якщо в ньому відкинути всі члени починаючи з деякого. Таких частинних сум достатньо для розв'язання багатьох задач з аналізу, геометрії та механіки, зокрема тих, де необхідне локальне наближення функції, тобто її наближення в малому околі деякої точки. Проте у задачах, де потрібно наближати функцію на деякій фіксованій множині точок (зокрема, на відрізку), частинні суми n-го порядку ряду Тейлора є не найкращими для наближення.

У даній роботі при наближенні степеневої функції функції $y = (1 + x)^{\alpha}$ многочленами n-го степеня використано A-метод, розроблений В. К. Дзяди-ком [1]. Суть цього методу така.