НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
НТУУ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

Освітньо-кваліфікаційний рівень магістр

Напрям підготовки 121 – Інженерія програмного забезпечення Семестр \_1\_

(назва)

Навчальна дисципліна Сучасні технології розроблення програмного забезпечення - 1

\_ (назва)

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 15

1. Принципи та підходи кросплатформеної розробки. Source-to-source compiler (transpiler). Переваги та недоліки, приклади використання.

**ПІДХОДИ:**

*1. Архітектурний підхід*

Виносимо все, що залежить від платформи, в окремі компоненти

Різні версії компонентів для різних платформ, Але однаковий програмний інтерфейс

Весь інший код – однаковий для різних платформ

ЯКЩО різні платформи на різних мовах програмування, то є такі підходи:

*а. Трансляція / трансформація в цільову мову*

Описуємо структуру та логіку застосунку на одній мові

Перетворюємо її в іншу мову

*б. Віртуальне середовище виконання*

Код, написаний на певній мові, виконується у віртуальному середовищі

Механізми взаємодії між цим середовищем та платформою

Обмежена взаємодія сприяє безпеці, але негативно впливає на продуктивність

*в. Поєднання кількох мов в одному проекті*

Пишемо різні компоненти на різних мовах/технологіях

Механізми взаємодії між ними

Проблеми з підтримкою

Що робити, коли єдиний розробник на NewExoticScript™ пішов у відпустку чи звільнився з роботи?

**Переваги та недоліки різних підходів**

*Архітектурний підхід* – ізоляція специфічних компонентів

*Трансляція/трансформація в цільову мову*

*Віртуальне середовище виконання - о*бмежена взаємодія сприяє безпеці, але негативно впливає на продуктивність

*Поєднання кількох мов в одному проекті -* Проблеми з підтримкою, складність в налагодженні механізмів взаємодії (можлива несумісність, конфлікти)

*Source-to-source compiler (transpiler)*

- Автоматичне перетворення коду з однієї мови на іншу

- Працює на рівні source code

- Часто використовується для реалізації

прототипів компіляторів нових мов

- Часто враховує особливості мови, але не

виклики бібліотек/фреймворків

**transpiler** - тип компілятора, який використовує вихідний код програми, написаної на одній мові програмування, в якості вихідних даних і виробляє еквівалентний вихідний код на іншій мові програмування. Транспайлер - «перекладач» для мов програмування, які працюють приблизно на одному і тому ж рівні абстракції; традиційний компілятор «переводить» з більш високого рівня мови програмування на мову більш низького рівня. Наприклад, транспайлер може виконати переклад програми з Паскаль на Сі.

2. Бібліотека/фреймворк React. Основні можливості, архітектура, встановлення та налаштування, приклади використання. Переваги та недоліки.

ReactJs – это JS фреймворк, представленный разработчиками Facebook. Существует много разнящихся между собой мнений по поводу пользы и целесообразности использования данного продукта. В преимуществах и недостатках ReactJs мы и предлагаем разобраться.

**Архитектура**

**О свойствах**

Сравнивать ReactJs с Angular или другими MVC фреймворками не имеет смысла, так как ReactJs – это только представление. React – это язык шаблонов в сочетании с несколькими функциями, которые позволяют отрисовать HTML, т.е. результат работы React – это HTML.

ReactJs реализует концепции реактивного программирования: изменение результата работы зависит от состояния компонентов. Так, A = B + C, и результат A всегда будет зависеть от значений B и C.

ReactJs постоянно работает с DOM, перерисовывая его при изменении условий (та часть DOM, которую меняет ReactJs, называется компонентом). Ранее подобная практика сильно бы отразилась на производительности приложения, но разработчики ReactJs подошли к решению данного вопроса кардинально: они полностью переписали DOM на Javascript.

Важной особенностью ReactJs является использование JSX. Это надстройка на JS, позволяющая использовать про-XML синтаксис в Javascript коде. JSX – это сочетание javascript и html, которые в связке являются непривычным синтаксисом для большинства разработчиков. Стандартом считается разделение JS части от разметки, что усложняет слежение за изменениями HTML -> JS -> HTML. JSX позволяет видеть все процессы в одном месте, не отвлекаясь на сложности грамотного и валидного кода. После компиляции JSX получается чистый JS.

**О минусах**

С какими сложностями могут столкнуться разработчики, начинающие знакомство с ReactJS:

**• Сложная документация**

На сайте разработчика туториал разбросан по разным вкладкам, информация не структурирована. Однако если с ReactJS работает целая команда, эту проблему удается быстро решить. А изучать документацию рекомендуем на devdocs.io, на ресурсе информация представлена в более структурированном виде.

**• Не все стандартные браузеры поддерживают React**

Для преодоления этой проблемы предлагается использовать дополнительные плагины, например, библиотеку ES5-shim для поддержки IE8. Существуют и другие расширения для ReactJs, однако, учитывая вес фреймворка, их использование лучше свести к минимуму.

**• Большинство виджетов - свои**

ReactJs достаточно молодой фреймворк, поэтому все, даже стандартные виджеты вам придется прописывать заново. Нужен кастомный выпадающий список или lightbox? Для таких простых задач вам придется написать немало кода.

**О плюсах**

Разработка UI ведется на основе отдельных компонентов - за подобными проектами стоит будущее разработки.

С ReactJs удобно работать в команде, на нем приятно писать и легко тестировать.

**Установка/ настройка Реакта**

— Вам понадобится **Node >= 6** на вашем компьютере

— создаем начальный фронтенд каркас

npm install -g create-react-app

create-react-app my-app

cd my-app

npm start

Данный пример использует инструменты билда, такие как **Babel** и **webpack**, но работает с нулевой конфигурацией. Ее необходимо будет исправить под нужды вашего проекта.

Когда ваше приложение готово для разворачивания на продакшене, нужно запустить команду npm run build. Она создаст оптимизированный билд вашего приложения