UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
Departamento de Ciências de Computação

Lab Package

Comparação entre versões do ECMAScript 5 e ECMAScript 6

Jederson Donizete Zuchi (NUSP 9407156)
Naylor Garcia Bachiega (NUSP 5567669)

Prof. Dr. Paulo C. Masiero

SÃO CARLOS
2016

ÍNDICE

1. IDENTIFICAÇÃO

1.1 Título: Comparação entre versões do ECMAScript 5 e ECMAScript 6

1.2 Tema: Comparação da compreensão de algoritmos baseada na alteração de sintaxe do ECMAScript 6 é mais fácil do que a 5

1.3 Área Técnica: Desenvolvimento de Sistemas

1.4 Autor: Jederson D. Zuchi e Naylor G. Bachiega

1.5 Afiliação: ICMC-USP

1.6 Local: FATEC-Taquaritinga

1.7 Data: Mai/2016

2. INTRODUÇÃO

Linguagens de programação orientadas a objetos oferecem facilidades como abstração, encapsulamento de dados, herança e reutilização de códigos. Dessa forma, o princípio da definição de classe para um objeto é um das características proeminentes utilizadas nesse tipo de programação para alcançar seus benefícios (SCHREFL, 1988).

Esse tipo de linguagem tornou-se popular na década de 1980, porém o conceito foi criado um pouco antes, em 1960. Foi um novo modo de estruturar a informação e comunicação. Muitos dos conceitos que são associados com POO (programação orientada a objetos) foram desenvolvidos como parte de uma linguagem chamada Simula.

Por volta de 1970, Alan Kay (Xerox PARC) organizou um grupo de pesquisa dedicado em criar uma linguagem de fácil compreensão, chamada Smalltalk. A partir

disso, outras linguagens começaram a surgir, como por exemplo C++ e Java (BUDD, 2002).

Uma classe é um tipo abstrato de dados equipado com uma possível parcial implementação (MEYER, 1988). Classes de objetos são definidas como especializações de outras mais gerais e herdam as propriedades dessas classes (SCHREFL, 1988).

Na Figura 1 é mostrada a sintaxe usual de declaração de classes, utilizada na década de 80, largamente adotada na maioria das linguagens de POO.

```
classe VEÍCULO
    atributos:
        VeículoNúmero
        VelocidadeAtual
        VelocidadeMáxima
        Proprietário
    métodos:
        AumentarVelocidade
        DiminuirVelocidade
        Visualizar

classe CAMINHÃO
    especialização de: VEÍCULO
    atributos:
        Carga
        Proprietário
    métodos:
        Carregar
        Descarregar
        Visualizar
```

Figura 1 – Exemplo de declaração de classe.

No exemplo, há duas classes (Veículo e Caminhão). A classe Veículo (superclasse, mais geral), contém atributos e métodos e a classe Caminhão (subclasse), herda os atributos e métodos da classe que está especializando, além dos seus próprios atributos e métodos.

As Figuras 2 e 3 mostram os modelos de declaração de classe utilizados para as linguagens de programação C++ e Java, respetivamente ~\cite{Budd:02}.

```
class VEÍCULO {
    int VeículoNúmero;
    int VelocidadeAtual;
    int VelocidadeMáxima;
    string Proprietário;

    public:
        void AumentarVelocidade();
        void DiminuirVelocidade();
        void Visualizar();
}

class CAMINHÃO: public VEÍCULO {
    int Carga;
    string Proprietário;

    public:
        void Carregar();
        void Descarregar();
        void Visualizar();
}
```

Figura 2 – Exemplo de declaração de classe com C++.

```
class VEÍCULO {
    public int VeículoNúmero;
    public int VelocidadeAtual;
    public int VelocidadeMáxima;
    public String Proprietário;

    public AumentarVelocidade();
    public DiminuirVelocidade();
    public Visualizar();
}

class CAMINHÃO extends VEÍCULO {
    public int Carga;
    public String Proprietário;

    public void Carregar();
    public void Descarregar();
    public void Visualizar();
}
```

Figura 3 – Exemplo de declaração de classe com C++.

Como já mencionado, essa sintaxe foi bem empregada em outras linguagens, diferentemente do ECMAScript (ES). O ES teve sua primeira edição em 1997, sendo baseada em linguagens bem conhecidas, como JavaScript (Netscape) e JScript (Microsoft). Foi inventada por Brendan Eich na Netscape e incorporada no Navigator 2.0 e no Internet Explorer 3.0.

Originalmente foi desenvolvida como uma linguagem de script (que é utilizada para manipular, customizar e automatizar facilidades em um sistema operacional), porém também é utilizada como uma linguagem de propósito (ECMA, 2015).

A Figura 4 mostra o modelo de declaração de classe utilizado para a linguagem de programação ES na versão (ECMA, 2015).

```
var VEÍCULO = function() {
    this.VeículoNúmero;
    this.VeloicadeAtual;
    this.VelocidadeMáxima;
    this.Proprietário;
}

VEÍCULO.prototype.AumentarVelocidade_
    = function();

VEÍCULO.prototype.DiminuirVelocidade_
    = function();

VEÍCULO.prototype.Visualizar_
    = function();

var CAMINHÃO = function() {
    this.Carga;
    this.Proprietário;
}

CAMINHÃO.prototype = Object.create_
    (VEÍCULO.prototype);

CAMINHÃO.prototype.Carregar_
    = function();

CAMINHÃO.prototype.Descarregar_
    = function();

CAMINHÃO.prototype.Visualizar_
    = function();
```

Figura 4 – Exemplo de declaração de classe com ES 5.

Na Figura 5, o Comitê Técnico 39 da ECMA (TC39), que mantém o padrão de linguagem ECMAScript alterou a sintaxe da definição de classe. Os membros da comissão são compostos pelos principais fabricantes de navegadores e outras empresas interessadas na Web. Como é possível notar, até a versão 5, a TC39 adotou um padrão de declaração de classe diferente do modelo comumente utilizado por outras linguagens.

```
class VEÍCULO {
    constructor () {
        this.VeículoNúmero;
        this.VelocidadeAtual;
        this.VelocidadeMáxima;
        this.Proprietário;
    }

    AumentarVelocidade();
    DiminuirVelocidade();
    Visualizar();
}

class CAMINHÃO extends VEÍCULO
    constructor() {
        super();
        this.Carga;
        this.Proprietário;
    }

    Carregar();
    Descarregar();
    Visualizar();
}
```

Figura 5 – Exemplo de declaração de classe com ES 6.

Na versão 6 é visível a diferença no modelo de declaração, aproximando dos modelos já conhecidos nas linguagens C++ e Java. Para investigar a eficiência dessa mudança de sintaxe, foi conduzido um experimento com participantes, com conhecimento das duas versões, para verificar a compreensão e eficiência dos modelos de declaração.

3. CARACTERIZAÇÃO

3.1 Tipo: In Vitro

3.2 Domínio: Sistemas de Informação

3.3 Linguagem: Português

3.4 Parceiros: ICMC-USP

3.5 Links:

3.6 Estimativa de realização:

3.7 Estimativa do número de replicações: 1

3.8 Glossário

4. DEFINIÇÃO DO ESTUDO EXPERIMENTAL

4.1 Objetivo do Estudo: Avaliar a compreensão dos estudantes na leitura de classes escritas com códigos em JavaScript nas versões do ECMAScript 5 e 6.

4.2 Objetivo Global: Avaliar a eficácia da compreensão da nova sintaxe de classe do ECMAScript em relação a versão anterior.

4.3 Objetivos Específicos:

Analise.....: Versão do ECMAScript 5 e 6 .

Com o proposito de.....: Avaliar.

A respeito da.....: Compreensão do código.

Do ponto de vista do...: Programador.

No contexto.....: Estudantes de graduação que queiram aprender Javascript.

4.4 Enfoque na Qualidade: Facilidade de compreensão do algoritmo.

4.5 Contexto: O estudo será conduzido em um ambiente acadêmico com estudantes de graduação que serão os sujeitos do experimento. Cada sujeito receberá um questionário para avaliar o grau de compreensão das classes escritas em JavaScript nas versões do ECMAScript 5 e 6.

4.6 Questões e Métricas:

4.6.1 Questões:

- Os estudantes conseguirão entender o que o código pretende fazer mais facilmente na versão 6 que não versão 5?
- Os estudantes levarão menos tempo para resolver o código da versão 6 que na versão 5?

4.6.2 Métricas:

- **Respostas Corretas:** Serão contadas as respostas corretas e incorretas dos estudantes para medir a eficácia em cada uma das questões apresentadas nas duas versões.
- **Tempo de Resposta:** Serão coletados os tempos de resposta para medir a eficiência do novo padrão.

4.7 Questões que não podem ser respondidas pelo experimento estudado:

4.8 Questões Abertas:

- O quanto é efetiva a metodologia utilizada nesse estudo?
- Os estudantes são adequados para serem sujeitos desse experimento?

5. PLANEJAMENTO

5.1 Formulação da Hipótese:

- **01':** Os estudantes compreenderão melhor as classes utilizando o ECMAScript 6 ao ECMAScript 5?

H0: Não haverá diferença de compreensão das classes.

Ha: A versão do código produzido pela ECMAScript 6 permitirá melhor compreensão que a versão 5.

- **02':** Os estudantes levarão mais tempo para resolver as questões sobre as classes do ECMAScript 6 ao ECMAScript 5?

H0: Não haverá diferença substancial no tempo de resposta entre as versões.

Ha: Haverá diferença substancial no tempo de resposta entre as versões.

5.2 Seleção das Variáveis:

5.2.1 Variáveis Independentes

- **Conhecimento sobre ECMAScript 5;**
- **Conhecimento sobre ECMAScript 6;**
- **Classe ECMAScript 5;**
- **Classe ECMAScript 6;**
- **Conhecimento sobre Orientação a Objetos:** verifica se o sujeito tem conhecimento de orientação a objetos e programação funcional.
- **Semestre:** qual o semestre que o aluno está cursando na graduação.
- **ClassFirst:** qual a classe que foi apresentada primeiro para o sujeito.

5.2.2 Variáveis dependentes

- **Número de respostas:** número de respostas corretas e incorretas.
- **Tempo:** tempo gasto para resolver os códigos.

5.3 Design do Experimento:

- Comparação pareada e balanceada: Todos os alunos vão comparar os dois métodos de definição de classes. Para minimizar o efeito da ordem, do tratamento que aparece para o sujeito, os objetos serão distribuídos randomicamente (block design).
- um fator com dois tratamentos.

5.4 Tratamentos: classe do ECMAScript 5 e 6.

Sujeito	ECMAScript 5	ECMAScript 6
1	1	2
2	2	1
3	2	1
4	2	1
5	1	2
6	1	2

5.5 Seleção dos Sujeitos:

- Foram selecionados 37 estudantes de graduação do curso de computação (escolha por conveniência). Todos eles já tinham experiência de programação com a linguagem ECMAScript nas versões 5 e 6. Para efeitos de análise, foram incluídas perguntas no questionário demográfico para verificar o tempo de experiência do aluno em programação orientada a objetos, programação utilizando a versão 5 e 6 do ECMAScript.

5.6 Teste Estatístico e Coleta de Dados:

Para cada questão correta será contabilizado 1 ponto e nenhum ponto para questões incorretas. O test-t será utilizado para analisar os resultados da compreensão do código.

O tempo de resposta será medido para analisar a eficiência do novo padrão. A média será utilizada para apresentar os resultados.

5.7 Instrumentação:

- Artefatos:

    - Termo de Consentimento de participação (Anexo 1): formulário será disponibilizado online para o estudante antes do experimento.
    - Formulário de coleta de dados (Anexo 2): formulário será disponibilizado para o estudante preencher antes do experimento.
    - Formulário de coleta de resultados (Anexo 3): será disponibilizado online para o estudante.
    - Formulário de feedback (Anexo 4): será disponibilizado online para o estudante após o experimento.

5.8 Validade dos Resultados:

5.8.1 Validade de Construção

- Na construção do experimento, cuidados foram tomados na tentativa de mesclar exercícios com um grau baixo de dificuldade, sendo que a iniciati-

va era verificar o entendimento da sintaxe de classe e não avaliar o estudante em si. Esse grau baixo pode ter influenciado no resultado.

5.8.1 Validade Interna

- Apesar de utilizar exemplos com um baixo grau de dificuldade, houve mudança significativa no tempo de resolução. Esse tempo poderia aumentar ou diminuir de acordo com o grau de dificuldade, não sendo abordado nesse experimento.

5.8.2 Validade Externa

- O estudo será executado em sala de aula na universidade e os sujeitos não possuem experiência profissional. Dessa forma, as conclusões desse estudo talvez não possam ser direcionadas para áreas da indústria.
- Um pequeno número de sujeitos participará do estudo. Sendo assim, é possível que qualquer resultado desse estudo seja em função da quantidade de participantes.

6. TREINAMENTO

6.1 Definição e Procedimento:

O treinamento será executado antes da instrumentação, exemplificando para o aluno como deverá ser conduzido o experimento, com no máximo 15 minutos de explicação. Após, um formulário online de treinamento será disponibilizado com a duração de 15 minutos para resolver o exemplo.

6.2 Instrutor: Jederson D. Zuchi

6.3 Participantes: 37 alunos de graduação.

6.4 Artefatos: formulário de treinamento online.

7. EXECUÇÃO

O treinamento e o experimento foram realizados em um ambiente controlado, com supervisão de um dos autores envolvidos nesse trabalho, para se certificar de que os estudantes executassem as tarefas sem ajuda externa.

Os estudantes foram incentivados a dar a resposta o mais rápido possível. A razão para esta escolha é simular uma sessão de codificação realista. Além disso, os indivíduos podem passar o tempo todo da experiência em uma única tarefa. O tempo disponível foi fixado em 5 minutos para cada exercício. Na prática, as estimativas foram bem conservadoras, visto que nenhum dos participantes utilizou o tempo total e, no formulário de feedback, os estudantes apontaram que o tempo foi adequado. Em qualquer caso, para evitar que a atenção diminuísse, a experiência foi desenhada para não demorar mais que uma hora (treinamento e o experimento).

O treinamento foi executado antes da instrumentação, exemplificando para o aluno como deverá ser conduzido o experimento, com no máximo 15 minutos de explicação. Após, um formulário online de treinamento foi disponibilizado com a duração de 15 minutos para resolver o exemplo.

Na fase de treinamento, foram dados um exemplo de classe do ES 5 e outro do ES 6. O instrutor forneceu as orientações sobre o ambiente, apresentou o modelo de exercício e tirou as dúvidas no prazo estipulado. Os alunos forneceram um feedback sobre esta fase, em que 37.5% e 31.2% dos sujeitos consideraram, respectivamente, excelente e boa a fase de treinamento. Os demais sujeitos, 31.2% consideraram regular, porém não especificaram o motivo. Todavia, não foi informado para os estudantes que deveriam comparar uma classe com outra, justamente para evitar um viés. As classes foram apresentadas sem especificar a versão na qual pertenciam e em ordem aleatória.

Após a fase de treinamento, preenchimento do termo de consentimento e questionário demográfico, os estudantes foram liberados para executar o experimento.

Os dados quantitativos foram analisados para testar as hipóteses apresentadas nesse trabalho. Foi utilizado o test-t como abordagem, visto que o conjunto de dados foi equilibrado. O tempo de resposta foi medido para analisar a eficiência do novo padrão. A média foi utilizada para apresentar os resultados.

7.1 Análise dos Resultados

Os dados foram verificados quanto à consistência. Dentre os 37 participantes do experimento, 1 teve problema com o computador (a máquina travou) durante a aplicação do formulário. Porém, o aluno trocou de máquina e recomeçou o teste, finalizando por completo.

Após os estudantes finalizarem os testes, o banco de dados foi verificado para checar se todas as respostas foram gravadas com sucesso. Por algum motivo desconhecido, 5 sujeitos tiveram seus registros de tempo sem valores. Nesse caso, estes estudantes foram excluídos das estatísticas.

7.2 Sumário das Estatísticas

Na Tabela 2 é possível verificar os 32 sujeitos (S) válidos para as estatísticas utilizadas nesse experimento, com a quantidade de respostas corretas (R) por versão, o tempo gasto para resolver cada exercício (E), a média (M) e o desvio padrão (DP).

Os totais e as médias das respostas corretas para as duas versões foram, coincidentemente, as mesmas, porém é visível que as médias dos tempos das respostas para os exercícios da versão 5 foram maiores que as médias da versão 6.

7.3 Informação Demográfica

Os 32 participantes responderam perguntas sobre seu conhecimento na linguagem ECMAScript nas versões 5 e 6 e em linguagens orientadas a objetos, conforme pode ser observado na Figura 6.

Tabela 2: Respostas e tempo dos sujeitos

	ES5				ES6			
S	R	E1	E2	E3	R	E1	E2	E3
1	3	83	36	109	2	46	32	71
2	2	46	94	124	2	29	32	32
3	2	60	56	35	3	34	98	36
4	3	158	52	77	2	26	48	99
5	2	93	23	49	3	31	16	20
6	3	44	103	71	2	43	26	59
7	2	104	49	53	3	33	31	23
8	2	38	42	62	2	47	22	16
9	1	94	40	35	1	27	9	7
10	3	42	82	75	2	26	27	99
11	3	70	53	45	3	34	14	38
12	2	90	66	32	3	17	72	11
13	2	44	20	21	1	50	12	34
14	2	67	40	74	2	45	11	17
15	3	71	57	88	3	19	83	154
16	3	84	32	33	2	39	16	13
17	3	79	53	62	3	55	24	14
18	3	54	66	219	3	56	13	23
19	3	107	96	124	3	42	24	74
20	2	133	166	166	2	47	73	74
21	3	131	248	204	3	80	76	185
22	0	130	14	32	1	21	14	8
23	3	73	123	97	3	44	25	37
24	1	65	106	94	2	43	41	39
25	1	126	34	13	0	43	48	14
26	1	189	35	29	1	23	21	13
27	3	169	162	132	3	35	84	66
28	2	173	173	122	2	29	40	46
29	3	75	54	58	3	47	34	40
30	2	92	124	128	3	46	31	41
31	3	68	93	43	3	57	24	26
32	3	24	8	84	3	19	35	14
M	2.3	89.9	75.0	80.9	2.3	38.5	36.1	45.1
DP	0.8	42.3	54.1	51.2	0.8	13.7	24.4	41.5

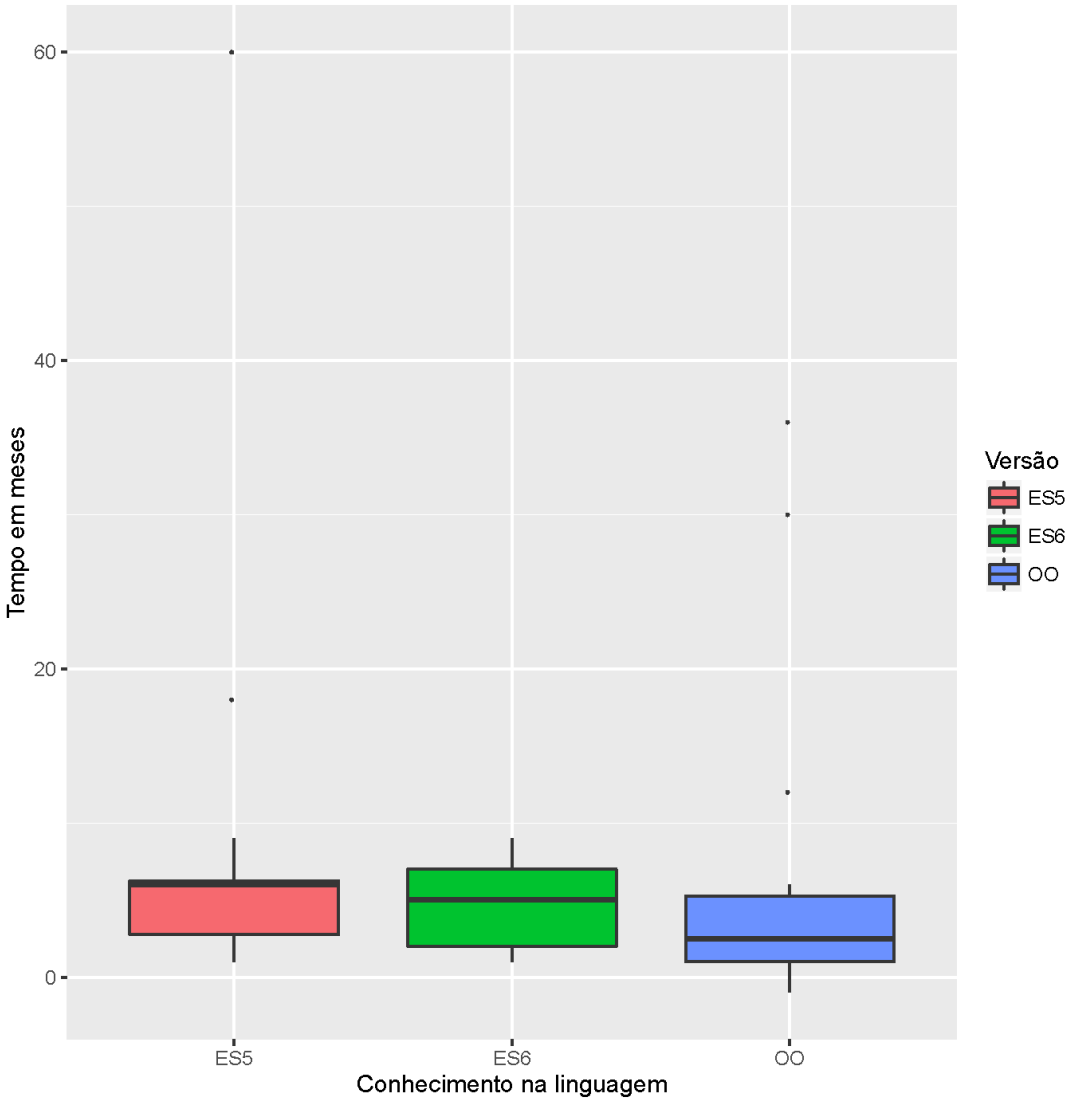

Figura 6 – Tempo de experiência.

Na Figura 7 é possível visualizar as respostas dos estudantes, em relação ao semestre corrente na graduação.

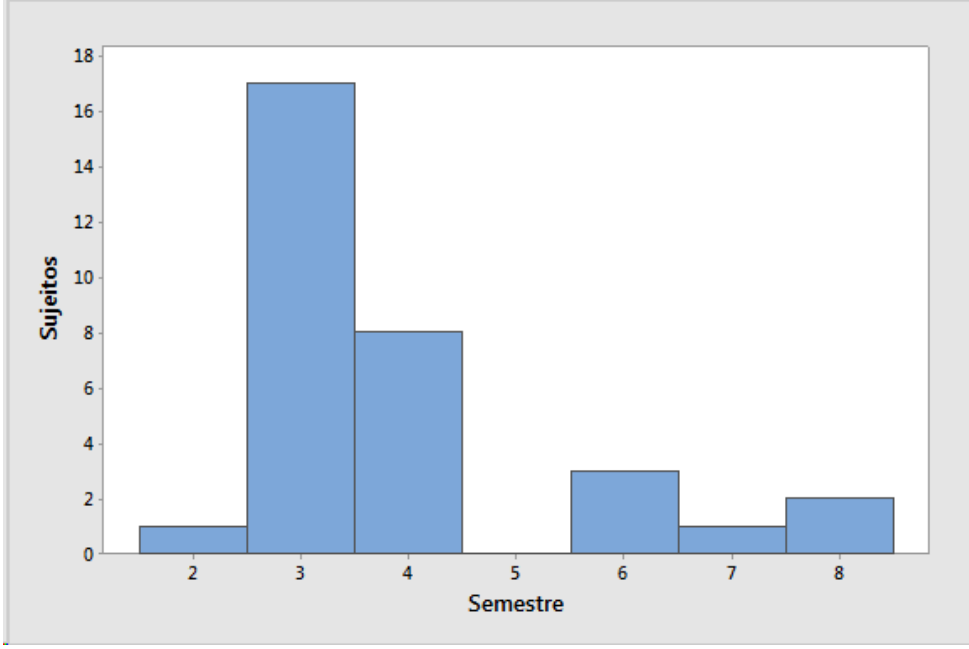

Figura 7 – Semestre.

7.4 Teste de Hipótese

A distribuição dos dados para ambas as hipóteses exibem uma distribuição normal, tanto para as respostas corretas (R) e o tempo (T), conforme Tabela 3. Testando a normalidade com o teste de Shapiro-Wilk e Kolmogorov-Smirnov tem-se um nível de significância inferior a 0,05 (GHASEMI, 2012).

Tabela 3: Teste de normalidade

	Kolmogorov-Smirnova			Shapiro-Wilk		
	Estatística	df	Sig.	Estatística	df	Sig.
R ES5	.299	32	.000	.777	32	,000
R ES6	.299	32	.000	.777	32	.000
T ES5	.189	32	.005	.866	32	.001
T ES6	.166	32	.024	.850	32	.000

7.4.1 Hipótese 1

Para a primeira hipótese, levando em consideração a distribuição normal dos dados e uma correlação positiva ($r=0.665$, $p<0.05$), a Tabela 4 mostra os resultados para o test-t pareado, como a média (M), desvio padrão (DP), o erro

do desvio padrão (EDP), o intervalo de confiança da diferença (ICD) e a significância.

Tabela 4: Test-t Pareado - Respostas

			95% ICD				
M	DP	EDP	Min.	Max	t	df	Sig.
.000	.672	.119	-.242	.242	,000	31	1.000

Como pode ser observado, não há diferença significativa entre as duas linguagens em relação às respostas corretas, isso fica evidente no histograma apresentado na Figura 8. Dessa forma, a hipótese nula, em relação às respostas corretas, não pode ser rejeitada.

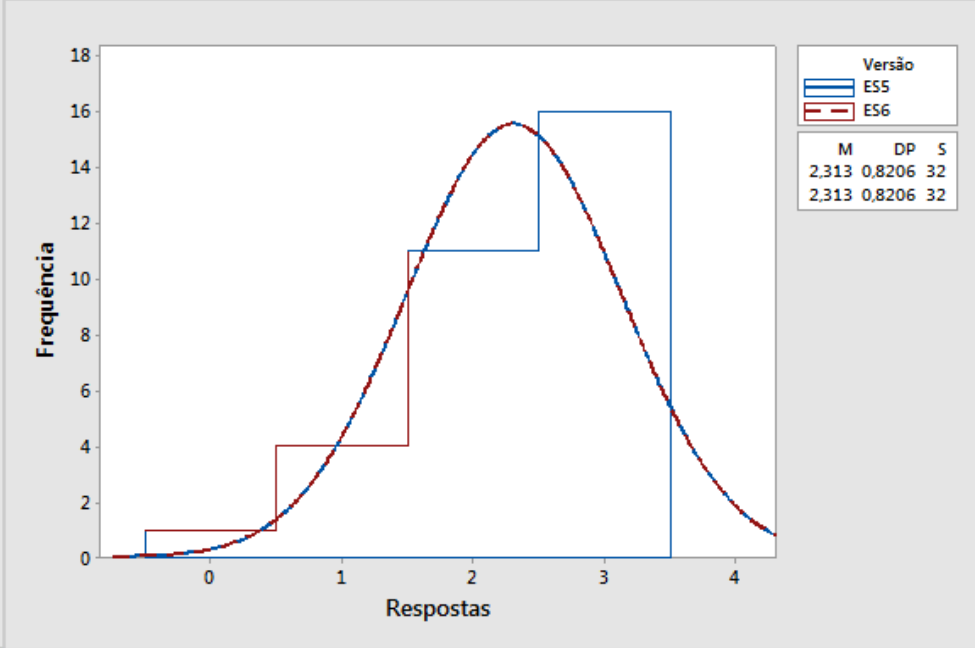

Figura 8 – Histograma para as respostas das versões ES5 e ES6.

7.4.2 Hipótese 2

Na segunda hipótese, levando em consideração a distribuição normal dos dados e uma correlação positiva ($r=0.629$, $p<0.05$), a Tabela 5 mostra os resultados para o test-t pareado.

Tabela 5: Test-t Pareado - Tempo

95% ICD							
M	DP	EDP	Min.	Max	t	df	Sig.
126.06	89.97	15.9	93.63	158.5	7.93	31	.000

Como pode ser observado, nesse caso, há diferença significativa entre as duas linguagens em relação ao tempo de resposta considerando todas as questões, isso fica evidente no histograma apresentado na Figura 9. Dessa forma, a hipótese nula, em relação ao tempo de resposta, pode ser rejeitada ($p<0.05$).

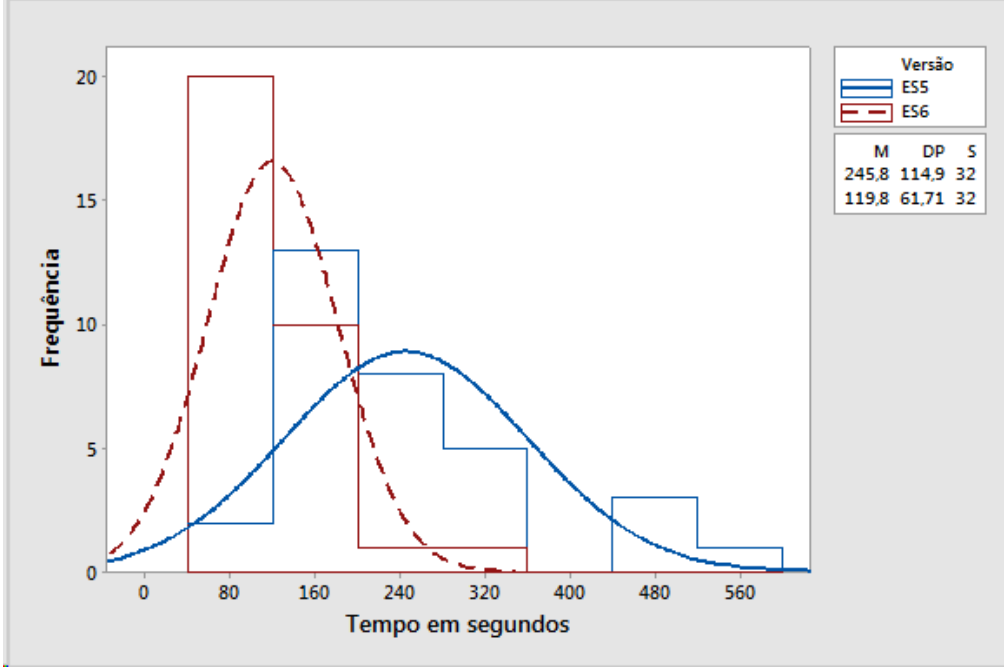

Figura 9 – Histograma para o tempo de resposta das versões ES5 e ES6.

Também foram realizados testes entre cada tipo de exercício aplicado no experimento, para verificar se há alguma relevância nas diferentes modalidades aplicadas, como sintaxe, herança e métodos get e set.

A Tabela 6 aponta que para o exercício 1 (sintaxe) há uma correlação negativa, porém com baixa significância ($p=0.537$). No exercício 2 (herança) tem-se correlação positiva de 0,466 ($p=0.07$) e o exercício 3 (get e set) apresenta uma correlação de 0.520 ($p=0.02$).

Tabela 6: Correção entre o tempo dos exercícios

	S	Correlação	Sig.
E1S5 - E1S6	32	-.113	.537
E2S5 - E2S6	32	.466	.007
E3S5 - E3S6	32	.520	.002

Na Figura 10, é possível observar a relação de tempo de cada tipo de exercício. A versão ES6 apresenta uma média de tempo menor que a versão ES5.

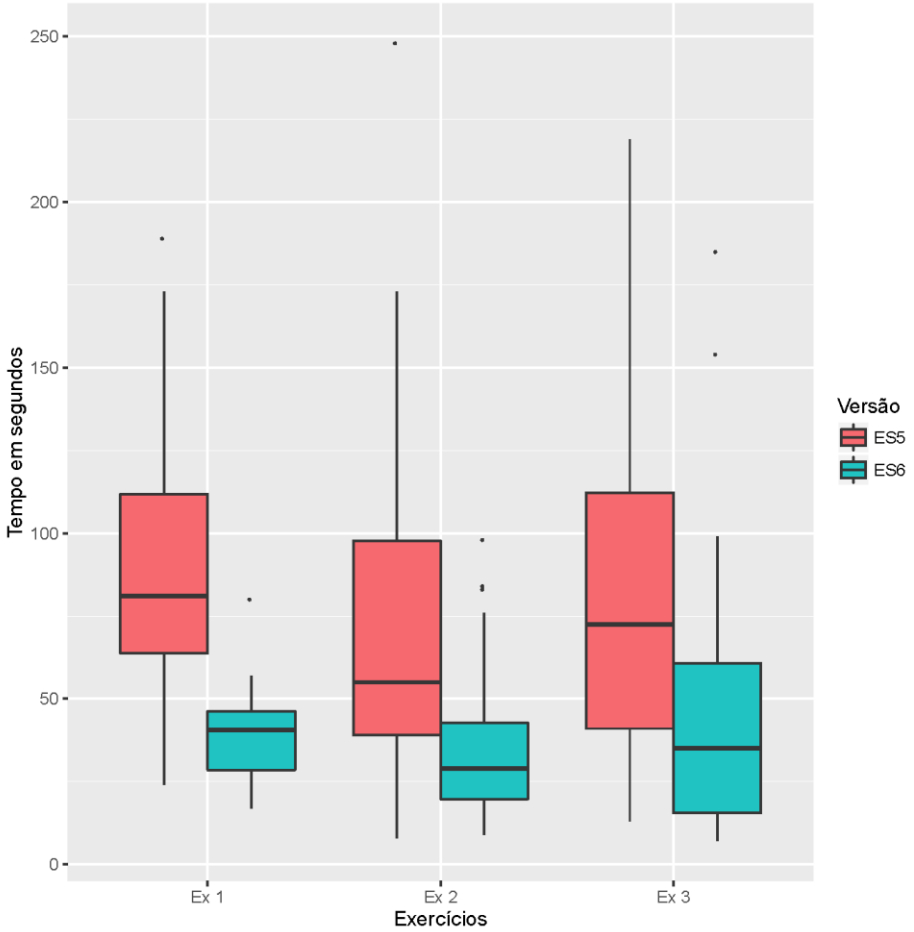

Figura 10 – Box plot do tempo de cada tipo de exercício.

Na Tabela 7, são apresentados os resultados para o test-t pareado para cada par de exercício das versões 5 e 6 do ECMAScript. Como já mencionado, nesse caso a hipótese nula pode ser rejeitada.

Tabela 7: Test-t Pareado - Tempo de cada exercício

				95% ICD				
	M	SD	SDE	Min	Max	t	df	Sig.
E1	51.34	45.85	8.10	34.81	67.87	6.33	31	.000
E2	38.87	47.84	8.45	21.62	56.12	4.59	31	.000
E3	35.84	46.14	8.15	19.20	52.48	4.39	31	.000

A Tabela 6 aponta que para o exercício 1 (sintaxe) há uma correlação negativa, porém com baixa significância (p=0.537). No exercício 2 (herança) tem-se correlação positiva de 0,466 (p=0.07) e o exercício 3 (get e set) apresenta uma correlação de 0.520 (p=0.02).

8. LIMITAÇÕES

8.1 Validade de Construção

A validade de construção refere-se a que o experimento faz para medir o que a teoria diz que faz. Ameaças nesse tipo de construção podem referir-se a fatores sociais, humanos ou definições incorretas no projeto (WOHLIN, 2012).

Na construção do experimento, cuidados foram tomados na tentativa de mesclar exercícios com um grau baixo de dificuldade, sendo que a iniciativa era verificar o entendimento da sintaxe de classe e não avaliar o estudante em si. Esse grau baixo pode ter influenciado no resultado.

8.2 Validade Interna

Esse tipo de validade diz respeito as relações de causa e efeito, ou seja, se um fator investigado é afetado por outro fator ou se este fator é afetado por um terceiro fator (WOHLIN, 2012).

Apesar de utilizar exemplos com um baixo grau de dificuldade, houve mudança significativa no tempo de resolução. Esse tempo poderia aumentar ou diminuir de acordo com o grau de dificuldade, não sendo abordado nesse experimento.

8.3 Validade Externa

Esse tipo de validade analisa até que ponto é possível generalizar os resultados, e em que medida os resultados são de interesse para outras pessoas fora do experimento realizado (WOHLIN, 2012).

O estudo será executado em sala de aula na universidade e os sujeitos não possuem experiência profissional. Dessa forma, as conclusões desse estudo talvez não possam ser direcionadas para áreas da indústria.

9. CONCLUSÃO

Apesar dos mesmos resultados para respostas corretas nas versões do ECMAScript, isso pode ter ocorrido pelo fato dos estudantes conhecerem as linguagens, porém o tempo para resolução dos exercícios evidencia a melhora na sintaxe da versão 6. Para trabalhos futuros, uma nova experiência com alunos que conhecem apenas orientação a objetos poderia ser realizada com esse propósito, assim como mudanças nos graus de dificuldades dos exercícios.

10. REFERÊNCIAS

Badreddin, O., Forward, A., and Lethbridge, T. C. (2012). **Model oriented programming: an empirical study of comprehension.** In Proceedings of the 2012 Conference of the Center for Advanced Studies on Collaborative Research, pages 73–86, NJ, USA. IBM Corp. Riverton.

Budd, T. A. (2002). **An Introduction to Object-Oriented Programming Third Edition.** Addison Wesley Longman. Ecma (2015). ECMAScript 2015 Language Specification. Ecma International. Engelschall, R. S. (2016). ECMAScript 6 — New Features: Overview & Comparison.

C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell and A. Wesslén. **Experimentation in Software Engineering.** Springer, ISBN 978-3-642-29043-5, 2012.

ECMA, International. **ECMAScript® 2015 Language Specification**. Disponível em: <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>. Acesso em: 10 abr. 2015.

Ghasemi, A. and Zahediasl, S. (2012). **Normality Tests for Statistical Analysis: A Guide for Non-Statisticians.** Int J Endocrinol Metab, 10:486–489.

Hanenberg, S. (2010). **An experiment about static and dynamic type systems: doubts about the positive impact of static type systems on development time.** In Proceedings of the ACM international conference on Object oriented programming systems languages and applications, pages 22–35, New York, NY, USA. ACM.

Lee, H., Won, S., Jin, J., Cho, J., and Ryu, S. (2005). **Safe: Formal specification and implementation of a scalable analysis framework for ECMAscript.** In In

Proceedings of the 2012 International Workshop on Foundations of Object-Oriented Languages. ACM.

Lopez-Nataren, C. and Viso-Gurovich, E. (2005). **An ecmascript compiler for the .net framework.** In Proceedings of the sixth Mexican international conference on computer science, pages 235–239. IEEE Computer Society.

Meyer, B. (1988). **Object-oriented software construction.** Prentice Hall.

Salvaneschi, G., Amann, S., Proksch, S., and Mezini, M. (2014). **An empirical study on program comprehension with reactive programming.** In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, pages 564–575, New York, NY, USA. ACM.

Schrefl, M. and Neuhold, E. J. (1988). **Object class definition by generalization using upward inheritance.** In Proceedings of the Fourth International Conference on Data Engineering, pages 4–13. IEEE Computer Society.

ANEXO 1

TERMO DE CONSENTIMENTO DE PARTICIPAÇÃO

Nome: _____(nome do aluno)_____

Este é um convite especial para você participar voluntariamente do estudo: "Estudo de comparação do ECMAScript 5 e 6". Por favor, leia com atenção as informações abaixo antes de dar seu consentimento para participar do estudo.

OBJETIVO E BENEFÍCIOS DO ESTUDO

O objetivo do estudo é verificar se as alterações propostas pela nova versão do ECMAScript 6, realmente irão facilitar o entendimento de códigos JavaScript em relação a versão 5.

PROCEDIMENTOS

O procedimento envolve o acesso a um site, em que o sujeito deverá analisar códigos escritos em JavaScript e dar o seu parecer sobre qual foi o código, em que ele considerou ser mais fácil o entendimento.

PARTICIPAÇÃO VOLUNTÁRIA

Sua participação neste estudo é ***voluntária*** e você terá plena e total liberdade para desistir do estudo a qualquer momento, sem que isso acarrete qualquer prejuízo.

GARANTIA DE SIGILO E PRIVACIDADE

As informações relacionadas ao estudo são confidenciais e qualquer informação divulgada em relatório ou publicação será feita sob forma codificada, para que a confidencialidade seja mantida. O pesquisador garante que seu nome não será divulgado sob hipótese alguma.

Diante do exposto acima eu, _____, declaro que fui esclarecido sobre os objetivos, procedimentos e benefícios do presente estudo. Participo de livre e espontânea vontade do estudo em questão. Foi-me assegurado o direito de abandonar o estudo a qualquer momento, se eu assim o desejar. Declaro também não possuir nenhum grau de dependência profissional ou educacional com os pesquisadores envolvidos nesse projeto (ou seja, os pesquisadores desse projeto não podem me prejudicar de modo algum no trabalho ou nos estudos), não me sentindo pressionado de nenhum modo a participar dessa pesquisa.

ANEXO 2

FORMULÁRIO DE COLETA DE DADOS

Nome:_____

Semestre corrente da graduação: _____

Questões Gerais

1. Possui alguma experiência com a linguagem de programação ECMAScript 5?

   Resposta: _____ (tempo em meses)

2. Possui alguma experiência com a linguagem de programação ECMAScript 6?

   Resposta: _____ (tempo em meses)

3. Possui alguma experiência com linguagens orientadas a objetos?

   Resposta: _____ (tempo em meses)

ANEXO 3

FORMULÁRIO DE COLETA DOS RESULTADOS

Nome:_____

Semestre corrente da graduação: _____

Analise as classes abaixo e determine o valor correto para cada uma:

Exemplo Classe X	Exemplo Classe Y
Classe...	Classe...
Resposta:	Resposta:
Exemplo Classe X	Exemplo Classe Y
Classe...	Classe...
Resposta:	Resposta:
Exemplo Classe X	Exemplo Classe Y
Classe...	Classe...
Resposta:	Resposta:

ANEXO 4

FORMULÁRIO DE FEEDBACK

Nome:_____

Semestre corrente da graduação: _____

Questões Gerais

1. A orientação sobre a realização do estudo antes da sua participação foi (marque a opção abaixo):

Ruim	Regular	Boa	Excelente

2. O tempo estabelecido para preenchimento do formulário foi:

Adequado	Não adequado

3. Caso desejar, informe abaixo suas considerações sobre o estudo:
