

Análise de ferramentas de diff

Oi pessoal!

Este formulário busca avaliar a eficácia de diferentes ferramentas de *diff* textual e sintático. A pesquisa tem como objetivo identificar qual ferramenta apresenta os resultados mais adequados em cada um dos cenários propostos. Os participantes da pesquisa devem ter, pelo menos, 6 meses de experiência como desenvolvedor de software e ter mais de 18 anos.

O formulário leva cerca de 15 minutos para ser respondido.

A participação de vocês é fundamental para o sucesso da minha pesquisa.

Muito obrigada pela colaboração!

DECLARAÇÃO DE CONSENTIMENTO

Ao responder a este questionário, você concorda em permitir que os pesquisadores responsáveis, utilizem e divulguem as informações **anônimas** fornecidas, conforme descrito abaixo.

Durante o estudo, há risco de confidencialidade e vazamento de dados, pois os participantes irão fornecer respostas sobre preferências no ambiente empresarial. Para mitigar isso, **os dados serão anonimizados**. Existe também a possibilidade de desconforto psicológico ao responder sobre práticas de trabalho, o que será minimizado com um formulário claro, objetivo, de no máximo 15 minutos, evitando questões controversas e sobrecarregamento.

A pesquisa não trará benefícios diretos aos participantes, mas seus resultados poderão futuramente contribuir para a melhor compreensão e uso das ferramentas estudadas, orientar programadores e pesquisadores, direcionar estudos acadêmicos e melhorar o processo de identificação de mudanças no código, tornando-o mais claro e preciso.

O projeto foi submetido e aprovado pelo Comitê de Ética em Pesquisa - CEP (CAAE: 84860524.2.0000.5208).

CONDIÇÕES E ESTIPULAÇÕES

1. Eu compreendo que todas as informações são confidenciais. Não serei identificado(a) pessoalmente. Concordo em completar o questionário para fins de pesquisa e que os dados derivados deste questionário **anônimo** podem ser publicados em revistas, conferências e postagens de blog.

2. Eu compreendo que minha participação neste questionário de pesquisa é **totalmente voluntária** e que se recusar a participar não implica em penalidade ou perda de benefícios.

3. Eu compreendo que posso contatar o pesquisador caso tenha qualquer dúvida sobre o questionário de pesquisa. Estou ciente de que meu consentimento não me trará benefícios diretos. Também estou ciente de que o autor manterá os dados coletados de forma perpétua e poderá utilizá-los em trabalhos acadêmicos futuros.

4. Eu compreendo que ao clicar no botão abaixo, confirmo que sou maior de idade e tenho mais de 18 anos completos.

5. Eu compreendo que os dados não podem ser removidos depois do preenchimento, mesmo que o participante solicite, visto que as informações não são identificáveis.

6. Ao clicar no botão abaixo, forneço livremente meu consentimento e reconheço meus direitos como participante voluntário de pesquisa, conforme descrito acima, e autorizo os pesquisadores a utilizarem minhas informações para a realização da pesquisa. Também confirmo que sou maior de idade para preencher este questionário.

Sua experiência

3. Há quanto tempo você trabalha como programador? * *Marcar apenas uma oval.*

- ☐ 6 meses - 1 ano
- ☐ 2 anos - 5 anos
- ☐ 5 anos ou mais

4. Você já trabalhou com alguma dessas linguagens de programação?

- ☐ Java
- ☐ C
- ☐ Outro:

5. Diariamente, utilizamos o comando "*git diff*" para mostrar a diferença entre versões de arquivos de código-fonte, este comando utiliza do mecanismo de *diff textual* para realizar este processo. No entanto, existem outras formas de realizar a diferenciação de arquivos e diretórios, como o *diff sintático*. Você já ouviu falar de ferramentas que utilizam o *diff*

sintático?

- ☐ Sim
- ☐ Não
- ☐ Já ouvir falar, mas nunca utilizei

Cenário 1

Resultado 1

```
@@ -1,14 +1,14 @@
-void Chunk_copy(Chunk *src, size_t src_start, Chunk *dst, size_t dst_start, size_t n)
+int Chunk_bounds_check(Chunk *chunk, size_t start, size_t n)
{
-   if (!Chunk_bounds_check(src, src_start, n)) return;
-   if (!Chunk_bounds_check(dst, dst_start, n)) return;
+   if (chunk == NULL) return 0;
-   memcpy(dst->data + dst_start, src->data + src_start, n);
+   return start <= chunk->length && n <= chunk->length - start;
}

-int Chunk_bounds_check(Chunk *chunk, size_t start, size_t n)
+void Chunk_copy(Chunk *src, size_t src_start, Chunk *dst, size_t dst_start, size_t n)
{
-   if (chunk == NULL) return 0;
+   if (!Chunk_bounds_check(src, src_start, n)) return;
+   if (!Chunk_bounds_check(dst, dst_start, n)) return;
-   return start <= chunk->length && n <= chunk->length - start;
+   memcpy(dst->data + dst_start, src->data + src_start, n);
}
\ No newline at end of file
```

Legenda do Resultado 1

Na imagem, é exibido o resultado da diferenciação entre duas versões, uma mais antiga e outra mais recente, de um mesmo arquivo. Cada linha indica uma ação identificada pela ferramenta:

- 1) Linha verde: Adição da respectiva linha no arquivo mais recente.
- 2) Linha vermelha: Deleção da respectiva linha no arquivo mais recente.

Resultado 2

```
testV1.c
1 void Chunk_copy(Chunk *src, size_t src_start, Chunk *dst, size_t dst_start, size_t n)
2 {
3     if (!Chunk_bounds_check(src, src_start, n)) return;
4     if (!Chunk_bounds_check(dst, dst_start, n)) return;
5     memcpy(dst->data + dst_start, src->data + src_start, n);
6 }
7
8 int Chunk_bounds_check(Chunk *chunk, size_t start, size_t n)
9 {
10     if (chunk == NULL) return 0;
11     return start <= chunk->length && n <= chunk->length - start;
12 }
13
14
testV2.c
1 int Chunk_bounds_check(Chunk *chunk, size_t start, size_t n)
2 {
3     if (chunk == NULL) return 0;
4     return start <= chunk->length && n <= chunk->length - start;
5 }
6
7 void Chunk_copy(Chunk *src, size_t src_start, Chunk *dst, size_t dst_start, size_t n)
8 {
9     if (!Chunk_bounds_check(src, src_start, n)) return;
10    if (!Chunk_bounds_check(dst, dst_start, n)) return;
11    memcpy(dst->data + dst_start, src->data + src_start, n);
12 }
13
14
```

Legenda do Resultado 2

Na imagem, exibe-se as duas versões de um mesmo arquivo, ao lado esquerdo a mais antiga e ao lado direito a mais recente.

Cada cor sinaliza uma ação identificada pela ferramenta:

- 1) Roxo: Movimentação do trecho de código (código existia em uma posição diferente no arquivo antigo)
- 2) Amarelo: Modificação de um elemento sintático
- 3) Verde: Adição de um novo trecho de código
- 4) Vermelho: Deleção de um trecho de código

6. Em uma situação de desenvolvimento, qual dos dois *diffs* você preferiria?

- ☐ Resultado 1
- ☐ Resultado 2
- ☐ Os dois resultados são equivalentes

7. Com base na sua resposta anterior, por favor, explique o que influenciou sua escolha. Se você marcou um dos dois como superior, o que considera ser a principal vantagem? Caso tenha marcado como equivalente, o que faz você ver as duas opções como similares?

8. Você achou que alguma das imagens era confusa ou difícil de entender?

- ☐ Sim, a imagem do Resultado 1
- ☐ Sim, a imagem do Resultado 2
- ☐ Sim, ambas são confusas
- ☐ Não, ambas são claras

9. A ação exemplificada na imagem (movimentação de código) costuma ocorrer regularmente em sua rotina de trabalho?

- ☐ Sim
- ☐ Não

Cenário 2

Resultado 3

```
4
5 import com.bittercode.constant.ErrorCodes;
6
7 public class StoreException extends IOException {
8
9     private String errorCode;
10    private String errorMessage;
11    private int statusCode;
12
13    public StoreException(String errorMessage) {
14        super(errorMessage);
15        this.errorCode = "BAD_REQUEST";
16        this.setStatusCode(400);
17        this.errorMessage = errorMessage;
18    }
19
20    public StoreException(ErrorCodes errorCodes) {
21        super(errorCodes.getMessage());
22        this.statusCode = errorCodes.getCode();
23        this.errorMessage = errorCodes.getMessage();
24        this.setErrorCode(errorCodes.name());
25    }
26 }
```

Legenda do Resultado 3

Na imagem, exibe-se as 2 versões de um mesmo arquivo, ao lado esquerdo a mais antiga e ao lado direito a mais recente.

Cada cor sinaliza uma ação identificada pela ferramenta:

- 1) Roxo: Movimentação do trecho de código
- 2) Amarelo: Modificação de um elemento sintático
- 3) Verde: Adição de um novo trecho de código
- 4) Vermelho: Deleção de um trecho de código

Resultado 4

```
index a4d527d..e4350ea 100644
--- a/src/main/java/com/bittercode/model/StoreException.java
+++ b/src/main/java/com/bittercode/model/StoreException.java
@@ -2,7 +2,7 @@ package com.bittercode.model;

import java.io.IOException;

-import com.bittercode.constant.ErrorCodes;
+import com.bittercode.constant.ResponseCode;

public class StoreException extends IOException {

@@ -17,7 +17,7 @@ public class StoreException extends IOException {
    this.errorMessage = errorMessage;
}

-    public StoreException(ErrorCodes errorCodes) {
+    public StoreException(ResponseCode errorCodes) {
        super(errorCodes.getMessage());
        this.statusCode = errorCodes.getCode();
        this.errorMessage = errorCodes.getMessage();
    }
}
```

Legenda do Resultado 4

Na imagem, é exibido o resultado da diferenciação entre duas versões, uma mais antiga e outra mais recente, de um mesmo arquivo. Cada linha indica uma ação identificada pela ferramenta:

- 1) Linha verde: Adição da respectiva linha no arquivo mais recente.
- 2) Linha vermelha: Deleção da respectiva linha no arquivo mais recente.

10. Em uma situação de desenvolvimento, qual dos dois *diffs* você preferiria?

- ☐ Resultado 3
- ☐ Resultado 4
- ☐ Os dois resultados são equivalentes

11. Com base na sua resposta anterior, por favor, explique o que influenciou sua escolha. Se você marcou um dos dois como superior, o que considera ser a principal vantagem? Caso tenha marcado como equivalente, o que faz você ver as duas opções como similares?

12. Você achou que alguma das imagens era confusa ou difícil de entender?

- ☐ Sim, a imagem do Resultado 3
- ☐ Sim, a imagem do Resultado 4
- ☐ Sim, ambas são confusas
- ☐ Não, ambas são claras

13. A ação exemplificada na imagem (mudança no import) costuma ocorrer regularmente em sua rotina de trabalho?

- ☐ Sim
- ☐ Não

Cenário 3

Resultado 5

```
103 @RequestMapping(value = "newuserregister", method = RequestMethod.POST)
104 public String newUserRegister(@ModelAttribute User user)
105 {
106     System.out.println(user.getEmail());
107     user.setRole("ROLE_NORMAL");
108     this.userService.addUser(user);
109     return "redirect:/";
110 }
111
112 //for Learning purpose of model
113 @GetMapping("/test")
114 public String Test(Model model)
115 {
116     System.out.println("test page");
117     model.addAttribute("author", "jay gajera");
118     model.addAttribute("id", 40);
119 }
120
121
122
123
```

```
104 @RequestMapping(value = "newuserregister", method = RequestMethod.POST)
105 public ModelAndView newUserRegister(@ModelAttribute User user)
106 {
107     // Check if username already exists in database
108     boolean exists = this.userService.checkUserExists(user.getUsername());
109
110     if (!exists) {
111         System.out.println(user.getEmail());
112         user.setRole("ROLE_NORMAL");
113         this.userService.addUser(user);
114
115         System.out.println("New user created: " + user.getUsername());
116         ModelAndView mView = new ModelAndView("userLogin");
117         return mView;
118     } else {
119         System.out.println("New user not created - username taken: " + user.ge
120         ModelAndView mView = new ModelAndView("register");
121         mView.addObject("msg", user.getUsername() + " is taken. Please choos
122         return mView;
123     }
124 }
```

Legenda do Resultado 5

Na imagem, exibe-se as 2 versões de um mesmo arquivo, ao lado esquerdo a mais antiga e ao lado direito a mais recente.

Cada cor sinaliza uma ação identificada pela ferramenta:

- 1) Roxo: Movimentação do trecho de código
- 2) Amarelo: Modificação de um elemento sintático
- 3) Verde: Adição de um novo trecho de código
- 4) Vermelho: Deleção de um trecho de código

Resultado 6

```

+ @RequestMapping(value = "newuserregister", method = RequestMethod.POST)
+ public String newUserRegister(@ModelAttribute User user)
+ public ModelAndView newUserRegister(@ModelAttribute User user)
+ {
+
+     System.out.println(user.getEmail());
+     user.setRole("ROLE_NORMAL");
+     this.userService.addUser(user);
+
+     return "redirect:/";
+     // Check if username already exists in database
+     boolean exists = this.userService.checkUserExists(user.getUsername());
+
+     if(!exists) {
+         System.out.println(user.getEmail());
+         user.setRole("ROLE_NORMAL");
+         this.userService.addUser(user);
+
+         System.out.println("New user created: " + user.getUsername());
+         ModelAndView mView = new ModelAndView("userLogin");
+         return mView;
+     } else {
+         System.out.println("New user not created - username taken: " + user.getUsername());
+         ModelAndView mView = new ModelAndView("register");
+         mView.addObject("msg", user.getUsername() + " is taken. Please choose a different username.");
+         return mView;
+     }
+ }

```

Legenda do Resultado 6

Na imagem, é exibido o resultado da diferenciação entre duas versões, uma mais antiga e outra mais recente, de um mesmo arquivo. Cada linha indica uma ação identificada pela ferramenta:

- 1) Linha verde: Adição da respectiva linha no arquivo mais recente.
- 2) Linha vermelha: Deleção da respectiva linha no arquivo mais recente.

14. Em uma situação de desenvolvimento, qual dos dois *diffs* você preferiria?

- ☐ Resultado 5
- ☐ Resultado 6
- ☐ Os dois resultados são equivalentes

15. Com base na sua resposta anterior, por favor, explique o que influenciou sua escolha. Se você marcou um dos dois como superior, o que considera ser a principal vantagem? Caso tenha marcado como equivalente, o que faz você ver as duas opções como similares?

16. Você achou que alguma das imagens era confusa ou difícil de entender?

- ☐ Sim, a imagem do Resultado 5
- ☐ Sim, a imagem do Resultado 6
- ☐ Sim, ambas são confusas
- ☐ Não, ambas são claras

17. A ação exemplificada na imagem (mudanças internas em um método) costuma ocorrer regularmente em sua rotina de trabalho?

- ☐ Sim
- ☐ Não

Cenário 4

Resultado 7

```
--- a/JtProject/src/main/java/com/jtspringproject/JtSpringProject/controller/AdminController.java
+++ b/JtProject/src/main/java/com/jtspringproject/JtSpringProject/controller/AdminController.java
@@ -81,7 +81,7 @@ public class AdminController {

    User user=this.userService.checkLogin(username, pass);

-    if(user.getRole().equals("ROLE_ADMIN")) {
+    if(user.getRole() != null && user.getRole().equals("ROLE_ADMIN")) {
        ModelAndView mv = new ModelAndView("adminHome");
        adminlogcheck=1;
        mv.addObject("admin", user);
    }
```

Legenda do Resultado 7

Na imagem, é exibido o resultado da diferenciação entre duas versões, uma mais antiga e outra mais recente, de um mesmo arquivo. Cada linha indica uma ação identificada pela ferramenta:

- 1) Linha verde: Adição da respectiva linha no arquivo mais recente.
- 2) Linha vermelha: Deleção da respectiva linha no arquivo mais recente.

Resultado 8

```
if(user.getRole().equals("ROLE_ADMIN")) {  
    ModelAndView mv = new ModelAndView("adminHome");  
    adminlogcheck=1;  
    mv.addObject("admin", user);  
    return mv;  
}  
else {  
    ModelAndView mv = new ModelAndView("adminlogin");  
    mv.addObject("msg", "Please enter correct username and password");  
    return mv;  
}
```

```
if(user.getRole() != null && user.getRole().equals("ROLE_ADMIN")) {  
    ModelAndView mv = new ModelAndView("adminHome");  
    adminlogcheck=1;  
    mv.addObject("admin", user);  
    return mv;  
}  
else {  
    ModelAndView mv = new ModelAndView("adminlogin");  
    mv.addObject("msg", "Please enter correct username and password");  
    return mv;  
}
```

Legenda do Resultado 8

Na imagem, exibe-se as 2 versões de um mesmo arquivo, ao lado esquerdo a mais antiga e ao lado direito a mais recente.

Cada cor sinaliza uma ação identificada pela ferramenta:

- 1) Roxo: Movimentação do trecho de código
- 2) Amarelo: Modificação de um elemento sintático
- 3) Verde: Adição de um novo trecho de código
- 4) Vermelho: Deleção de um trecho de código

18. Em uma situação de desenvolvimento, qual dos dois *diffs* você preferiria?

- ☐ Resultado 7
- ☐ Resultado 8
- ☐ Os dois resultados são equivalentes

19. Com base na sua resposta anterior, por favor, explique o que influenciou sua escolha. Se você marcou um dos dois como superior, o que considera ser a principal vantagem? Caso tenha marcado como equivalente, o que faz você ver as duas opções como similares?

20. Você achou que alguma das imagens era confusa ou difícil de entender?

- ☐ Sim, a imagem do Resultado 7
- ☐ Sim, a imagem do Resultado 8
- ☐ Não, ambas são claras

21. A ação exemplificada na imagem (alteração de um condicional) costuma ocorrer regularmente em sua rotina de trabalho?

- ☐ Sim
- ☐ Não

22. Após analisar os casos, você acredita que o uso de diff sintático poderia impactar de alguma forma a sua rotina de trabalho? Comente sobre:

23. Você estaria interessado em adotar ferramentas de diff sintático em seus projetos futuros?

- ☐ Sim
- ☐ Não
- ☐ Talvez