

Доработка языка программирования Free Pascal: реализация замыканий

Выполнил студент группы с8503а

Кевролетин Василий Владимирович

Руководитель: старший преподаватель кафедры
информатики, математического и компьютерного
моделирования Кленин Александр Сергеевич

Дальневосточный Федеральный университет
2013г.

Технические особенности проекта

- Поддержка большого числа процессоров.
- Поддержка большого числа операционных систем.
- Поддержка нескольких диалектов Pascal.

Организационные особенности проекта

- Открытый исходный код.
- Разрабатывается постоянной командой добровольцев.
- Принимает доработки от сторонних разработчиков.

Понятие замыкания

Пример замыкания

```
1 type
2   TAccum=reference to function (n: Integer): Integer;
3
4 function MakeAccumulator: TAccum;
5 var value: Integer;
6 begin
7   value := 0;
8   result := function (n: Integer): Integer begin
9       value := value + n;
10      result := value;
11      end;
12 end;
```

Анонимные функции

Пример

```
1 function Factory: TProc;  
2 begin  
3     Result := procedure  
4         begin  
5             Writeln( 'executed ' );  
6         end;  
7 end;
```

Вложенные функции

Пример

```
1  procedure outer;  
2  var i: Integer;  
3  
4      procedure inner; begin  
5          i := 10;  
6      end;  
7  
8  begin  
9      ...  
10 end;
```

Пример: продление жизни локальных переменных

```
1 function Factory(data: Integer): TProc;  
2 begin  
3     Result := procedure  
4         begin  
5             Writeln( data );  
6         end;  
7     end;  
8  
9 var f1: TProc;  
10 begin  
11     f1 := Factory(10);  
12     f2 := Factory(20);  
13     f1 ();           { 10 }  
14     f2 ();           { 20 }  
15 end.
```

Пример

```
1  var i: Integer;  
2      fSet: TIntProc;  
3      fWrite: TProc;  
4  begin  
5      i := 0;  
6      fSet := procedure(n: Integer) begin  
7          i := n;  
8      end;  
9      fWrite := procedure begin  
10         Writeln(i);  
11     end;  
12     i := 10;  
13     fWrite();           { 10 }  
14     fSet(20);  
15     fWrite();           { 20 }  
16 end.
```

Пример

```
1 int main()  
2 {  
3     int i = 0;  
4     auto f = [=] { std::cout << i; };  
5     i = 10;  
6     f();           /* 0 */  
7 }
```


Реализация замыканий в современных ЯП

ЯП	Анонимные функции	Вложенные функции	Захват по значению	Захват по ссылке	Замыкания
Perl	+	+/-		+	+
Python	+	+		+	+
Ruby	+	+		+	+
Scheme	+	+		+	+
Elisp	+	+		+/-	
Scala	+	+		+	+
Java				+	+/-
C					
C++	+		+	+/-	+
Delphi	+	+		+	+
Fpc		+			

Сложности управления памятью

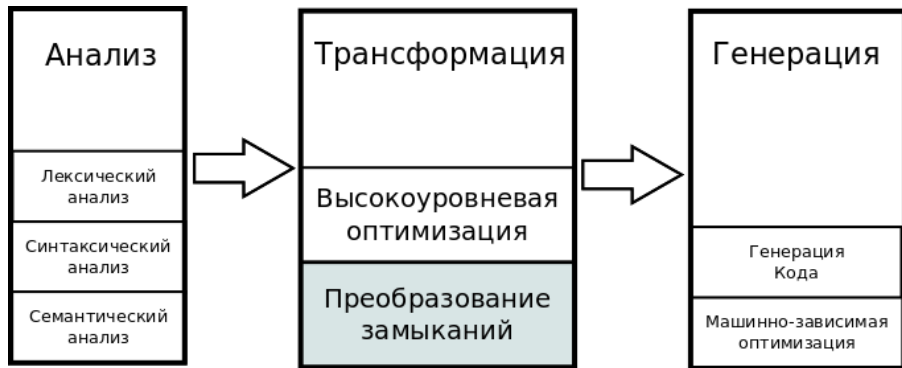
Несколько замыканий захватили одну переменную

```
1  var i : Integer;  
2  begin  
3    p1 := procedure begin Inc(i); end;  
4    p2 := procedure begin Dec(i); end;  
5  end.
```

Замыкание – фактический параметр функции

```
1  mapContainer.ForEach(  
2    procedure(key, val: String) begin  
3      sum := sum + val;  
4    end );
```

Структура компилятора



Выявление захваченных переменных

```
1  type
2      TProc = reference to
3              procedure;
4
5  var  d1, d2: Integer;
6      p: TProc;
7
8  begin
9
10     d1 := 0;
11     d2 := 0;
12     p := procedure begin
13         Inc(d1);
14         Dec(d2);
15     end;
16     p;
17 end.
```

Создание хранилища

```
1  type
2      TProc = reference to
3              procedure;
4
5  var  d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8  begin
9      store := TStore.Create;
10     d1 := 0;
11     d2 := 0;
12     p := procedure begin
13         Inc(d1);
14         Dec(d2);
15     end;
16     p;
17 end.
```

```
1  type
2
3
4
5
6      TStore = class
7          d1: Integer;
8          d2: Integer;
9      end;
10
11
12
13
14
15
16
17  //
```

Перенаправление доступа

```
1  type
2      TProc = reference to
3              procedure;
4
5  var d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8  begin
9      store := TStore.Create;
10     store.d1 := 0;
11     store.d2 := 0;
12     p := procedure begin
13         Inc(d1);
14         Dec(d2);
15     end;
16     p;
17 end.
```

```
1  type
2
3
4
5
6
7      TStore = class
8          d1: Integer;
9          d2: Integer;
10     end
11
12
13
14
15
16
17 //
```

Модификация анонимной функции

```
1  type
2      TProc = reference to
3              procedure;
4
5  var d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8  begin
9      store := TStore.Create;
10     store.d1 := 0;
11     store.d2 := 0;
12     p := procedure begin
13         Inc(d1);
14         Dec(d2);
15     end;
16     p;
17 end.
```

```
1  type
2
3
4
5
6
7      TStore = class
8          d1: Integer;
9          d2: Integer;
10         procedure Anonym;
11     end
12
13     procedure TStore.Anonym;
14     begin
15         Inc(d1);
16         Dec(d2);
17     end;
```

Перенаправление доступа

```
1  type
2      TProc = reference to
3              procedure;
4
5  var d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8  begin
9      store := TStore.Create;
10     store.d1 := 0;
11     store.d2 := 0;
12     p := procedure begin
13         Inc(d1);
14         Dec(d2);
15     end;
16     p;
17 end.
```

```
1  type
2
3
4
5
6      TStore = class
7          d1: Integer;
8          d2: Integer;
9          procedure Anonym;
10     end
11
12     procedure TStore.Anonym;
13     begin
14         Inc(self.d1);
15         Dec(self.d2);
16     end;
```


Замыкание – указатель на метод?

```
1  type
2      TProc = procedure of object;
3
4
5  var  d1, d2: Integer;
6        p: TProc;
7        store: TStore;
8  begin
9      store := TStore.Create;
10     store.d1 := 0;
11     store.d2 := 0;
12     p := @store.Anonym;
13
14
15     p();
16 end.
```

```
1  type
2
3
4
5
6      TStore = class
7          d1: Integer;
8          d2: Integer;
9          procedure Anonym;
10      end
11
12
13     procedure TStore.Anonym;
14     begin
15         Inc(self.d1);
16         Dec(self.d2);
17     end;
```

Замыкание – указатель на интерфейс

```
1  type
2      TProc = reference to
3              procedure;
4
5  var  d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8
9  begin
10     store := TStore.Create;
11     store.d1 := 0;
12     store.d2 := 0;
13     p := @store.Anonym;
14
15     p();
16
17 end.
```

```
1  type
2      TClosureIntf = interface
3          procedure Anonym;
4      end;
5
6      TStore =
7          class(TClosureIntf)
8              d1: Integer;
9              d2: Integer;
10             procedure Anonym;
11         end
12
13     procedure TStore.Anonym;
14     begin
15         Inc(self.d1);
16         Dec(self.d2);
17     end;
```

Вызов замыкания

```
1  type
2      TProc = interface
3          procedure Apply;
4      end;
5  var d1, d2: Integer;
6      p: TProc;
7      store: TStore;
8  begin
9      store := TStore.Create;
10     store.d1 := 0;
11     store.d2 := 0;
12     p :=
13         TClosureIntf(store);
14
15     p.Apply;
16 end.
```

```
1  type
2      TClosureIntf = interface
3          procedure Anonym;
4      end;
5
6      TStore =
7          class(TClosureIntf)
8              d1: Integer;
9              d2: Integer;
10             procedure Anonym;
11         end
12
13     procedure TStore.Anonym;
14     begin
15         Inc(self.d1);
16         Dec(self.d2);
17     end;
```

Проделанная работа

- Изучена предметная область.
- Создана реализация замыканий для компилятора Free Pascal.

Сложности с большим уровнем вложенности

Создание объекта

```
1  type
2    TProc = reference to procedure;
3    TFactory = reference to function: TProc;
4
5  function ProduceFactory: TFactory;
6  var
7    data: Integer=0;
8  begin
9    result := function: TProc
10      begin
11        result := procedure
12          begin
13            Inc(data);
14          end;
15      end;
16  end;
```