

Überflüssige Kommentare

```
/*  
 * default  
 */  
default:  
    break;  
  
/* return SUCCESS */  
return SUCCESS;  
  
zeroCount++: /* Increment zero entry counter */  
  
/* Initialize total to numberReceived. */  
Node.total = Node.numberReceived;
```

Überflüssige Kommentare

```
while ( ((c = getChar()) != EOF) && (isSpace(c)) )  
    ;                               /* skip white space */  
if (c == EOF)                       /* end of file */  
    type = END_OF_FILE;  
else if (c == '(')                  /* left paren */  
    type = LEFT_PAREN;  
else if (c == ')')                  /* right paren */  
    type = RIGHT_PAREN;  
else if (c == ';')                  /* semicolon */  
    type = SEMICOLON;  
else if (isOp(c))                   /* operator */  
    type = OPERATOR;  
else if (isDigit(c))               /* number */  
    type = NUMBER;  
...
```

Verschluckte (hicks) Exceptions

```
try
{
    FileOutputStream out = new FileOutputStream( strName );
    ...
    out.flush();
    out.close();
}
catch( IOException e )
{ ;
}
```

```
int f (int f[], int l, Item k)
{ int i;
  for (i=0; i < l; i++)
    if (f [i] == k) return i;
}
```

```
int f (int f[], int l, Item k)
{ int i;
  for (i=0; i < l; i++)
    if (f [i] == k) return i;
}
```

In der Kürze liegt die Würze (idKldW)?

```
int Index (int *field , int length , Item key)
{ int i;
  for (i=0; i < length; i++)
    if (field [i] == key) return i;
}
```

Software-Redundanz

foo.c

```
...  
for (i = 1; i < MAX; i++) {  
    x = func() * y + x;  
}  
...  
for (j = 1; j < MAX; j++) {  
    x = func() * y + x;  
}  
...
```

bar.c

```
...  
for (i = 1; i < MAX; i++) {  
    x = func() * y + x;  
}  
...
```

fred.c

```
...  
for (i = 1; i < MAX; i++) {  
    x1 = func() * y1 + x1;  
}  
...  
for (i = 1; i < MAX; i++) {  
    x2 = func() * y2 + x2;  
}  
...
```

Typisch: 5 – 30 % des Codes sind redundant. . .

Software-Redundanz

foo.c

```
...  
  
set_coordinate (&x, y);  
  
...  
  
set_coordinate (&x, y);  
  
...
```

bar.c

```
...  
  
set_coordinate (&x, y);  
  
...
```

```
void set_coordinate (  
    float *x, float y) {  
    int i;  
    for (i = 1; i < MAX; i++) {  
        *x = func() * y + *x;  
    }  
}
```

fred.c

```
...  
  
set_coordinate (&x1, y1);  
  
...  
  
set_coordinate (&x2, y2);  
  
...
```

... und können abstrahiert werden

File Edit Options Buffers Tools C++ Help

```
// This procedure uses a quick sort algorithm to sort the ERRORS
// by the left_line_no and left_column_no fields.
```

```
void ParseError::SortMessages()
{
    int lower,
        upper,
        lostack[32],
        histack[32];

    int top,
        i,
        j;
    ParseErrorInfo pivot, temp;

    top = 0;
    lostack[top] = 0;
    histack[top] = errors.Length() - 1;

    while (top >= 0)
    {
        lower = lostack[top];
        upper = histack[top];
        top--;

        while (upper > lower)
        {
            /* The array is most-likely almost sorted. Therefore,
            /* we use the middle element as the Pivot element.
            i = (lower + upper) / 2;
            pivot = errors[i];
            errors[i] = errors[lower];

            /* Split the array section indicated by LOWER and UPPER
            /* using ARRAY(LOWER) as the pivot.
            i = lower;
            for (j = lower + 1; j <= upper; j++)
            if ((errors[j].left_token < pivot.left_token) ||
            /* When two error messages start in the same location
            /* and one is nested inside the other, the outer one
            /* is placed first so that it can be printed last.
            /* Recall that its right-span location is reached
            /* after the inner one has been completely processed.
            (errors[j].left_token == pivot.left_token &&
            errors[j].right_token > pivot.right_token) ||
            /* When two error messages are at the same location
            /* span, check the NUM field to keep the sort stable.
            /* When the location spans only a single symbol,
            /* the one with the lowest "num" is placed first.
            (errors[j].left_token == pivot.left_token &&
            errors[j].right_token == pivot.right_token &&
```

```
// This procedure uses a quick sort algorithm to sort the ERRORS
// by the left_line_no and left_column_no fields.
```

```
void SemanticError::SortMessages()
{
    int lower,
        upper,
        lostack[32],
        histack[32];

    int top,
        i,
        j;
    ErrorInfo pivot,
        temp;

    top = 0;
    lostack[top] = 0;
    histack[top] = error.Length() - 1;

    while (top >= 0)
    {
        lower = lostack[top];
        upper = histack[top];
        top--;

        while (upper > lower)
        {
            /* The array is most-likely almost sorted. Therefore,
            /* we use the middle element as the pivot element.
            i = (lower + upper) / 2;
            pivot = error[i];
            error[i] = error[lower];

            /* Split the array section indicated by LOWER and
            /* using ARRAY(LOWER) as the pivot.
            i = lower;
            for (j = lower + 1; j <= upper; j++)
            if ((error[j].left_token < pivot.left_token) ||
            /* When two error messages start in the same location
            /* and one is nested inside the other, the outer one
            /* is placed first so that it can be printed last.
            /* Recall that its right-span location is reached
            /* after the inner one has been completely processed.
            (error[j].left_token == pivot.left_token &&
            error[j].right_token > pivot.right_token) ||
            /* When two error messages are at the same location
            /* span, check the NUM field to keep the sort stable.
            /* When the location spans only a single symbol,
            /* the one with the lowest "num" is placed first.
            (error[j].left_token == pivot.left_token &&
            error[j].right_token == pivot.right_token &&
```


Überlange Methoden

```

11  def get_neighbors(id: int) list int:
12      """Returns a list of neighbors of a given node, as
13      well as the number of neighbors it has.
14      """
15      neighbors = []
16      for i in range(1, 1000000):
17          if i % 1000000 == 0:
18              print('Checking node %d' % i)
19          neighbors.append(id + i)
20      return neighbors
21
22  def get_neighbors2(id: int) list int:
23      """Returns a list of neighbors of a given node, as
24      well as the number of neighbors it has.
25      """
26      neighbors = []
27      for i in range(1, 1000000):
28          if i % 1000000 == 0:
29              print('Checking node %d' % i)
30          neighbors.append(id + i)
31      return neighbors
32
33  def get_neighbors3(id: int) list int:
34      """Returns a list of neighbors of a given node, as
35      well as the number of neighbors it has.
36      """
37      neighbors = []
38      for i in range(1, 1000000):
39          if i % 1000000 == 0:
40              print('Checking node %d' % i)
41          neighbors.append(id + i)
42      return neighbors
43
44  def get_neighbors4(id: int) list int:
45      """Returns a list of neighbors of a given node, as
46      well as the number of neighbors it has.
47      """
48      neighbors = []
49      for i in range(1, 1000000):
50          if i % 1000000 == 0:
51              print('Checking node %d' % i)
52          neighbors.append(id + i)
53      return neighbors
54
55  def get_neighbors5(id: int) list int:
56      """Returns a list of neighbors of a given node, as
57      well as the number of neighbors it has.
58      """
59      neighbors = []
60      for i in range(1, 1000000):
61          if i % 1000000 == 0:
62              print('Checking node %d' % i)
63          neighbors.append(id + i)
64      return neighbors
65
66  def get_neighbors6(id: int) list int:
67      """Returns a list of neighbors of a given node, as
68      well as the number of neighbors it has.
69      """
70      neighbors = []
71      for i in range(1, 1000000):
72          if i % 1000000 == 0:
73              print('Checking node %d' % i)
74          neighbors.append(id + i)
75      return neighbors
76
77  def get_neighbors7(id: int) list int:
78      """Returns a list of neighbors of a given node, as
79      well as the number of neighbors it has.
80      """
81      neighbors = []
82      for i in range(1, 1000000):
83          if i % 1000000 == 0:
84              print('Checking node %d' % i)
85          neighbors.append(id + i)
86      return neighbors
87
88  def get_neighbors8(id: int) list int:
89      """Returns a list of neighbors of a given node, as
90      well as the number of neighbors it has.
91      """
92      neighbors = []
93      for i in range(1, 1000000):
94          if i % 1000000 == 0:
95              print('Checking node %d' % i)
96          neighbors.append(id + i)
97      return neighbors
98
99  def get_neighbors9(id: int) list int:
100     """Returns a list of neighbors of a given node, as
101     well as the number of neighbors it has.
102     """
103     neighbors = []
104     for i in range(1, 1000000):
105         if i % 1000000 == 0:
106             print('Checking node %d' % i)
107         neighbors.append(id + i)
108     return neighbors
109
110  def get_neighbors10(id: int) list int:
111     """Returns a list of neighbors of a given node, as
112     well as the number of neighbors it has.
113     """
114     neighbors = []
115     for i in range(1, 1000000):
116         if i % 1000000 == 0:
117             print('Checking node %d' % i)
118         neighbors.append(id + i)
119     return neighbors
120
121  def get_neighbors11(id: int) list int:
122     """Returns a list of neighbors of a given node, as
123     well as the number of neighbors it has.
124     """
125     neighbors = []
126     for i in range(1, 1000000):
127         if i % 1000000 == 0:
128             print('Checking node %d' % i)
129         neighbors.append(id + i)
130     return neighbors
131
132  def get_neighbors12(id: int) list int:
133     """Returns a list of neighbors of a given node, as
134     well as the number of neighbors it has.
135     """
136     neighbors = []
137     for i in range(1, 1000000):
138         if i % 1000000 == 0:
139             print('Checking node %d' % i)
140         neighbors.append(id + i)
141     return neighbors
142
143  def get_neighbors13(id: int) list int:
144     """Returns a list of neighbors of a given node, as
145     well as the number of neighbors it has.
146     """
147     neighbors = []
148     for i in range(1, 1000000):
149         if i % 1000000 == 0:
150             print('Checking node %d' % i)
151         neighbors.append(id + i)
152     return neighbors
153
154  def get_neighbors14(id: int) list int:
155     """Returns a list of neighbors of a given node, as
156     well as the number of neighbors it has.
157     """
158     neighbors = []
159     for i in range(1, 1000000):
160         if i % 1000000 == 0:
161             print('Checking node %d' % i)
162         neighbors.append(id + i)
163     return neighbors
164
165  def get_neighbors15(id: int) list int:
166     """Returns a list of neighbors of a given node, as
167     well as the number of neighbors it has.
168     """
169     neighbors = []
170     for i in range(1, 1000000):
171         if i % 1000000 == 0:
172             print('Checking node %d' % i)
173         neighbors.append(id + i)
174     return neighbors
175
176  def get_neighbors16(id: int) list int:
177     """Returns a list of neighbors of a given node, as
178     well as the number of neighbors it has.
179     """
180     neighbors = []
181     for i in range(1, 1000000):
182         if i % 1000000 == 0:
183             print('Checking node %d' % i)
184         neighbors.append(id + i)
185     return neighbors
186
187  def get_neighbors17(id: int) list int:
188     """Returns a list of neighbors of a given node, as
189     well as the number of neighbors it has.
190     """
191     neighbors = []
192     for i in range(1, 1000000):
193         if i % 1000000 == 0:
194             print('Checking node %d' % i)
195         neighbors.append(id + i)
196     return neighbors
197
198  def get_neighbors18(id: int) list int:
199     """Returns a list of neighbors of a given node, as
200     well as the number of neighbors it has.
201     """
202     neighbors = []
203     for i in range(1, 1000000):
204         if i % 1000000 == 0:
205             print('Checking node %d' % i)
206         neighbors.append(id + i)
207     return neighbors
208
209  def get_neighbors19(id: int) list int:
210     """Returns a list of neighbors of a given node, as
211     well as the number of neighbors it has.
212     """
213     neighbors = []
214     for i in range(1, 1000000):
215         if i % 1000000 == 0:
216             print('Checking node %d' % i)
217         neighbors.append(id + i)
218     return neighbors
219
220  def get_neighbors20(id: int) list int:
221     """Returns a list of neighbors of a given node, as
222     well as the number of neighbors it has.
223     """
224     neighbors = []
225     for i in range(1, 1000000):
226         if i % 1000000 == 0:
227             print('Checking node %d' % i)
228         neighbors.append(id + i)
229     return neighbors
230
231  def get_neighbors21(id: int) list int:
232     """Returns a list of neighbors of a given node, as
233     well as the number of neighbors it has.
234     """
235     neighbors = []
236     for i in range(1, 1000000):
237         if i % 1000000 == 0:
238             print('Checking node %d' % i)
239         neighbors.append(id + i)
240     return neighbors
241
242  def get_neighbors22(id: int) list int:
243     """Returns a list of neighbors of a given node, as
244     well as the number of neighbors it has.
245     """
246     neighbors = []
247     for i in range(1, 1000000):
248         if i % 1000000 == 0:
249             print('Checking node %d' % i)
250         neighbors.append(id + i)
251     return neighbors
252
253  def get_neighbors23(id: int) list int:
254     """Returns a list of neighbors of a given node, as
255     well as the number of neighbors it has.
256     """
257     neighbors = []
258     for i in range(1, 1000000):
259         if i % 1000000 == 0:
260             print('Checking node %d' % i)
261         neighbors.append(id + i)
262     return neighbors
263
264  def get_neighbors24(id: int) list int:
265     """Returns a list of neighbors of a given node, as
266     well as the number
```

[illegible]

Line #	Code	Comment
1	<code>def __init__(self, data):</code>	
2	<code>self.data = data</code>	
3	<code>self.index = 0</code>	
4	<code>self.next = None</code>	
5	<code>def __str__(self):</code>	
6	<code>return str(self.data) + ' -> ' + str(self.next)</code>	
7	<code>def __repr__(self):</code>	
8	<code>return self.__str__()</code>	
9	<code>def __len__(self):</code>	
10	<code>return 1 + len(self.next)</code>	
11	<code>def __getitem__(self, index):</code>	
12	<code>if index < 0:</code>	
13	<code> index = -index - 1</code>	
14	<code>if index < 0 or index > len(self):</code>	
15	<code> raise IndexError('Index out of range')</code>	
16	<code>if index == 0:</code>	
17	<code> return self.data</code>	
18	<code>else:</code>	
19	<code> return self.next[index]</code>	
20	<code>def __setitem__(self, index, value):</code>	
21	<code>if index < 0:</code>	
22	<code> index = -index - 1</code>	
23	<code>if index < 0 or index > len(self):</code>	
24	<code> raise IndexError('Index out of range')</code>	
25	<code>if index == 0:</code>	
26	<code> self.data = value</code>	
27	<code>else:</code>	
28	<code> self.next[index] = value</code>	
29	<code>def __delitem__(self, index):</code>	
30	<code>if index < 0:</code>	
31	<code> index = -index - 1</code>	
32	<code>if index < 0 or index > len(self):</code>	
33	<code> raise IndexError('Index out of range')</code>	
34	<code>if index == 0:</code>	
35	<code> self.data = None</code>	
36	<code>else:</code>	
37	<code> self.next[index] = None</code>	
38	<code>def __iter__(self):</code>	
39	<code>return self</code>	
40	<code>def __next__(self):</code>	
41	<code>if self.index < len(self):</code>	
42	<code> return self.data</code>	
43	<code>else:</code>	
44	<code> raise StopIteration</code>	
45	<code>def __add__(self, other):</code>	
46	<code>if not isinstance(other, list):</code>	
47	<code> raise TypeError('Can only add list to list')</code>	
48	<code>return self + list(other)</code>	
49	<code>def __sub__(self, other):</code>	
50	<code>if not isinstance(other, list):</code>	
51	<code> raise TypeError('Can only subtract list from list')</code>	
52	<code>return self - list(other)</code>	
53	<code>def __mul__(self, other):</code>	
54	<code>if not isinstance(other, list):</code>	
55	<code> raise TypeError('Can only multiply list by list')</code>	
56	<code>return self * list(other)</code>	
57	<code>def __div__(self, other):</code>	
58	<code>if not isinstance(other, list):</code>	
59	<code> raise TypeError('Can only divide list by list')</code>	
60	<code>return self / list(other)</code>	
61	<code>def __mod__(self, other):</code>	
62	<code>if not isinstance(other, list):</code>	
63	<code> raise TypeError('Can only take modulus of list by list')</code>	
64	<code>return self % list(other)</code>	
65	<code>def __divmod__(self, other):</code>	
66	<code>if not isinstance(other, list):</code>	
67	<code> raise TypeError('Can only take divmod of list by list')</code>	
68	<code>return self // list(other), self % list(other)</code>	
69	<code>def __pow__(self, other):</code>	
70	<code>if not isinstance(other, list):</code>	
71	<code> raise TypeError('Can only take power of list by list')</code>	
72	<code>return self ** list(other)</code>	
73	<code>def __radd__(self, other):</code>	
74	<code>if not isinstance(other, list):</code>	
75	<code> raise TypeError('Can only add list to list')</code>	
76	<code>return list(other) + self</code>	
77	<code>def __rsub__(self, other):</code>	
78	<code>if not isinstance(other, list):</code>	
79	<code> raise TypeError('Can only subtract list from list')</code>	
80	<code>return list(other) - self</code>	
81	<code>def __rmul__(self, other):</code>	
82	<code>if not isinstance(other, list):</code>	
83	<code> raise TypeError('Can only multiply list by list')</code>	
84	<code>return list(other) * self</code>	
85	<code>def __rdiv__(self, other):</code>	
86	<code>if not isinstance(other, list):</code>	
87	<code> raise TypeError('Can only divide list by list')</code>	
88	<code>return list(other) / self</code>	
89	<code>def __rmod__(self, other):</code>	
90	<code>if not isinstance(other, list):</code>	
91	<code> raise TypeError('Can only take modulus of list by list')</code>	
92	<code>return list(other) % self</code>	
93	<code>def __rdivmod__(self, other):</code>	
94	<code>if not isinstance(other, list):</code>	
95	<code> raise TypeError('Can only take divmod of list by list')</code>	
96	<code>return list(other) // self, list(other) % self</code>	
97	<code>def __pow__(self, other):</code>	
98	<code>if not isinstance(other, list):</code>	
99	<code> raise TypeError('Can only take power of list by list')</code>	
100	<code>return list(other) ** self</code>	

[illegible]

	Code	Comment
1	<code>if (isFinite(1 / (1 - α)))</code>	
2	<code> α = 1 / (1 - α);</code>	
3	<code> α = 1 - α;</code>	
4	<code> α = 1 / (1 - α);</code>	
5	<code> α = 1 - α;</code>	
6	<code> α = 1 / (1 - α);</code>	
7	<code> α = 1 - α;</code>	
8	<code> α = 1 / (1 - α);</code>	
9	<code> α = 1 - α;</code>	
10	<code> α = 1 / (1 - α);</code>	
11	<code> α = 1 - α;</code>	
12	<code> α = 1 / (1 - α);</code>	
13	<code> α = 1 - α;</code>	
14	<code> α = 1 / (1 - α);</code>	
15	<code> α = 1 - α;</code>	
16	<code> α = 1 / (1 - α);</code>	
17	<code> α = 1 - α;</code>	
18	<code> α = 1 / (1 - α);</code>	
19	<code> α = 1 - α;</code>	
20	<code> α = 1 / (1 - α);</code>	
21	<code> α = 1 - α;</code>	
22	<code> α = 1 / (1 - α);</code>	
23	<code> α = 1 - α;</code>	
24	<code> α = 1 / (1 - α);</code>	
25	<code> α = 1 - α;</code>	
26	<code> α = 1 / (1 - α);</code>	
27	<code> α = 1 - α;</code>	
28	<code> α = 1 / (1 - α);</code>	
29	<code> α = 1 - α;</code>	
30	<code> α = 1 / (1 - α);</code>	
31	<code> α = 1 - α;</code>	
32	<code> α = 1 / (1 - α);</code>	
33	<code> α = 1 - α;</code>	
34	<code> α = 1 / (1 - α);</code>	
35	<code> α = 1 - α;</code>	
36	<code> α = 1 / (1 - α);</code>	
37	<code> α = 1 - α;</code>	
38	<code> α = 1 / (1 - α);</code>	
39	<code> α = 1 - α;</code>	
40	<code> α = 1 / (1 - α);</code>	
41	<code> α = 1 - α;</code>	
42	<code> α = 1 / (1 - α);</code>	
43	<code> α = 1 - α;</code>	
44	<code> α = 1 / (1 - α);</code>	
45	<code> α = 1 - α;</code>	
46	<code> α = 1 / (1 - α);</code>	
47	<code> α = 1 - α;</code>	
48	<code> α = 1 / (1 - α);</code>	
49	<code> α = 1 - α;</code>	
50	<code> α = 1 / (1 - α);</code>	
51	<code> α = 1 - α;</code>	
52	<code> α = 1 / (1 - α);</code>	
53	<code> α = 1 - α;</code>	
54	<code> α = 1 / (1 - α);</code>	
55	<code> α = 1 - α;</code>	
56	<code> α = 1 / (1 - α);</code>	
57	<code> α = 1 - α;</code>	
58	<code> α = 1 / (1 - α);</code>	
59	<code> α = 1 - α;</code>	
60	<code> α = 1 / (1 - α);</code>	
61	<code> α = 1 - α;</code>	
62	<code> α = 1 / (1 - α);</code>	
63	<code> α = 1 - α;</code>	
64	<code> α = 1 / (1 - α);</code>	
65	<code> α = 1 - α;</code>	
66	<code> α = 1 / (1 - α);</code>	
67	<code> α = 1 - α;</code>	
68	<code> α = 1 / (1 - α);</code>	
69	<code> α = 1 - α;</code>	
70	<code> α = 1 / (1 - α);</code>	
71	<code> α = 1 - α;</code>	
72	<code> α = 1 / (1 - α);</code>	
73	<code> α = 1 - α;</code>	
74	<code> α = 1 / (1 - α);</code>	
75	<code> α = 1 - α;</code>	
76	<code> α = 1 / (1 - α);</code>	
77	<code> α = 1 - α;</code>	
78	<code> α = 1 / (1 - α);</code>	
79	<code> α = 1 - α;</code>	
80	<code> α = 1 / (1 - α);</code>	
81	<code> α = 1 - α;</code>	
82	<code> α = 1 / (1 - α);</code>	
83	<code> α = 1 - α;</code>	
84	<code> α = 1 / (1 - α);</code>	
85	<code> α = 1 - α;</code> </	

[illegible]

```

10  function get(id, id2) begin
11    get := get + 1
12    id2 := id + 1
13    id := id + 1
14    id := id + 1
15    id := id + 1
16    id := id + 1
17    id := id + 1
18    id := id + 1
19    id := id + 1
20    id := id + 1
21    id := id + 1
22    id := id + 1
23    id := id + 1
24    id := id + 1
25    id := id + 1
26    id := id + 1
27    id := id + 1
28    id := id + 1
29    id := id + 1
30    id := id + 1
31    id := id + 1
32    id := id + 1
33    id := id + 1
34    id := id + 1
35    id := id + 1
36    id := id + 1
37    id := id + 1
38    id := id + 1
39    id := id + 1
40    id := id + 1
41    id := id + 1
42    id := id + 1
43    id := id + 1
44    id := id + 1
45    id := id + 1
46    id := id + 1
47    id := id + 1
48    id := id + 1
49    id := id + 1
50    id := id + 1
51    id := id + 1
52    id := id + 1
53    id := id + 1
54    id := id + 1
55    id := id + 1
56    id := id + 1
57    id := id + 1
58    id := id + 1
59    id := id + 1
60    id := id + 1
61    id := id + 1
62    id := id + 1
63    id := id + 1
64    id := id + 1
65    id := id + 1
66    id := id + 1
67    id := id + 1
68    id := id + 1
69    id := id + 1
70    id := id + 1
71    id := id + 1
72    id := id + 1
73    id := id + 1
74    id := id + 1
75    id := id + 1
76    id := id + 1
77    id := id + 1
78    id := id + 1
79    id := id + 1
80    id := id + 1
81    id := id + 1
82    id := id + 1
83    id := id + 1
84    id := id + 1
85    id := id + 1
86    id := id + 1
87    id := id + 1
88    id := id + 1
89    id := id + 1
90    id := id + 1
91    id := id + 1
92    id := id + 1
93    id := id + 1
94    id := id + 1
95    id := id + 1
96    id := id + 1
97    id := id + 1
98    id := id + 1
99    id := id + 1
100   id := id + 1
101   id := id + 1
102   id := id + 1
103   id := id + 1
104   id := id + 1
105   id := id + 1
106   id := id + 1
107   id := id + 1
108   id := id + 1
109   id := id + 1
110   id := id + 1
111   id := id + 1
112   id := id + 1
113   id := id + 1
114   id := id + 1
115   id := id + 1
116   id := id + 1
117   id := id + 1
118   id := id + 1
119   id := id + 1
120   id := id + 1
121   id := id + 1
122   id := id + 1
123   id := id + 1
124   id := id + 1
125   id := id + 1
126   id := id + 1
127   id := id + 1
128   id := id + 1
129   id := id + 1
130   id := id + 1
131   id := id + 1
132   id := id + 1
133   id := id + 1
134   id := id + 1
135   id := id + 1
136   id := id + 1
137   id := id + 1
138   id := id + 1
139   id := id + 1
140   id := id + 1
141   id := id + 1
142   id := id + 1
143   id := id + 1
144   id := id + 1
145   id := id + 1
146   id := id + 1
147   id := id + 1
148   id := id + 1
149   id := id + 1
150   id := id + 1
151   id := id + 1
152   id := id + 1
153   id := id + 1
154   id := id + 1
155   id := id + 1
156   id := id + 1
157   id := id + 1
158   id := id + 1
159   id := id + 1
160   id := id + 1
161   id := id + 1
162   id := id + 1
163   id := id + 1
164   id := id + 1
165   id := id + 1
166   id := id + 1
167   id := id + 1
168   id := id + 1
169   id := id + 1
170   id := id + 1
171   id := id + 1
172   id := id + 1
173   id := id + 1
174   id := id + 1
175   id := id + 1
176   id := id + 1
177   id := id + 1
178   id := id + 1
179   id := id + 1
180   id := id + 1
181   id := id + 1
182   id := id + 1
183   id := id + 1
184   id := id + 1
185   id := id + 1
186   id := id + 1
187   id := id + 1
188   id := id + 1
189   id := id + 1
190   id := id + 1
1
```

[illegible][illegible][illegible][illegible][illegible]

Ignoranz, Not-Invented-Here-Syndrom oder die neuerfundenen Räder

```
public class Articles {  
    private Article item;  
    private Articles next;  
  
    public Articles() { ... }  
    public void Add (Article art) { ... }  
    public void Show () { ... }  
}
```

Ignoranz, Not-Invented-Here-Syndrom oder die neuerfundenen Räder

```
public class Articles {  
    private Article item;  
    private Articles next;  
  
    public Articles() { ... }  
    public void Add (Article art) { ... }  
    public void Show () { ... }  
}  
  
public class Articles {  
    private java.util.LinkedList articles;  
  
    public Articles() { ... }  
    public void Add (Article art) { ... }  
    public void Show () { ... }  
}
```

Die Oglala des 21. Jahrhunderts

```
public class Articles {  
    private java.util.LinkedList articles;  
  
    public Articles() { ... }  
  
    public void Add (Article art) { articles.addLast (art); }  
  
    public void Show () {  
        Listlterator listltr = articles.listlterator();  
        while(listltr.hasNext()) {  
            Article art = (Article)listltr.next();  
            art.show();  
        }  
    }  
}
```

Die Oglala des 21. Jahrhunderts

```
public class Articles {  
    private java.util.LinkedList articles;  
  
    public Articles() { ... }  
  
    public void Add (Article art) { articles.addLast (art); }  
  
    public void Show () {  
        Listlterator listltr = articles.listlterator();  
        while(listltr.hasNext()) {  
            Article art = (Article)listltr.next();  
            art.show();  
        }  
    }  
}
```

Oglala (bedeutet: „Die ihre Habe verschleudern“ im Sinne von Großzügigkeit) sind ein Stamm der Lakota-Sioux-Indianer.

- einheitlich
- so einfach wie möglich
- sprechend, direkt verständlich
- prägnant
- frei von Redundanz
- übersichtlich
- strukturiert
- abgeschlossen
- abstrakt
- Separation of Concerns

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...
- alle Methoden einer Klasse static; die statischen Variablen werden über den Konstruktor initialisiert

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...
- alle Methoden einer Klasse static; die statischen Variablen werden über den Konstruktor initialisiert
- alle Methoden einer Klasse static, die Instanz, auf der die Methoden arbeiten, werden z.B. als LinkedList jeweils übergeben

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...
- alle Methoden einer Klasse static; die statischen Variablen werden über den Konstruktor initialisiert
- alle Methoden einer Klasse static, die Instanz, auf der die Methoden arbeiten, werden z.B. als LinkedList jeweils übergeben
- Neuimplementierung von Sortieralgorithmen

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...
- alle Methoden einer Klasse static; die statischen Variablen werden über den Konstruktor initialisiert
- alle Methoden einer Klasse static, die Instanz, auf der die Methoden arbeiten, werden z.B. als LinkedList jeweils übergeben
- Neuimplementierung von Sortieralgorithmen
- die gesamte GUI wird in einer einzigen Klasse implementiert

Was alles in den Code-Inspektionen 2004/05 auffiel

- Verwendung von verketteten Listen, wo Maps angebracht wären; manuelle Implementierung der entsprechenden Map-Zugriffe
- Verwendung von verketteten Listen, wo ArrayList angebracht wäre; z.B. Iterieren durch die LinkedList über Indexzugriffe...
- alle Methoden einer Klasse static; die statischen Variablen werden über den Konstruktor initialisiert
- alle Methoden einer Klasse static, die Instanz, auf der die Methoden arbeiten, werden z.B. als LinkedList jeweils übergeben
- Neuimplementierung von Sortieralgorithmen
- die gesamte GUI wird in einer einzigen Klasse implementiert
- Datenbankzugriffe werden über alle Klassen verteilt, statt gekapselt

Was alles in den Code-Inspektionen 2004/05 auffiel

- Query wird mit `select *` gemacht, danach werden manuell die benötigten Spalten rausgefiltert

Was alles in den Code-Inspektionen 2004/05 auffiel

- Query wird mit `select *` gemacht, danach werden manuell die benötigten Spalten rausgefiltert
- Datenbanknamen usw. werden hartcodiert

Was alles in den Code-Inspektionen 2004/05 auffiel

- Query wird mit `select *` gemacht, danach werden manuell die benötigten Spalten rausgefiltert
- Datenbanknamen usw. werden hartcodiert
- manuelle Implementierung von Parsern für verschiedene Zwecke (wo z.B. XML oder Properties-Datei angebracht wäre)

Was alles in den Code-Inspektionen 2004/05 auffiel

- Query wird mit select * gemacht, danach werden manuell die benötigten Spalten rausgefiltert
- Datenbanknamen usw. werden hartcodiert
- manuelle Implementierung von Parsern für verschiedene Zwecke (wo z.B. XML oder Properties-Datei angebracht wäre)
- mehrfache Implementierung der gleichen Hilfsklassen von verschiedenen Leuten, z.B. Übersetzungen (dementsprechend auch mit verschiedenen Dateiformaten)

Was alles in den Code-Inspektionen 2004/05 auffiel

- Query wird mit select * gemacht, danach werden manuell die benötigten Spalten rausgefiltert
- Datenbanknamen usw. werden hartcodiert
- manuelle Implementierung von Parsern für verschiedene Zwecke (wo z.B. XML oder Properties-Datei angebracht wäre)
- mehrfache Implementierung der gleichen Hilfsklassen von verschiedenen Leuten, z.B. Übersetzungen (dementsprechend auch mit verschiedenen Dateiformaten)
- keine Berücksichtigung von Performance (z.B. bei Operationen auf verketteter Liste, Dateioperationen; Konfigurationsdatei wird bei jedem Methodenaufruf neu eingelesen)

2 Benutzerdokumentation

- Lernziele
- Was ist Benutzerdokumentation?
- Arten der Dokumentation
- Struktur
- Auf Papier oder elektronisch?
- Autor
- Wie schreibt man eine Dokumentation?
- Hinweise und Regeln

Eine Benutzerdokumentation erstellen können

- wissen, was hinein gehört
- Arten kennen
- ihre Qualitäten kennen
- Prozess ihrer Erstellung kennen

While you're waiting, read the free novel we sent you. It's a Spanish story about a guy named 'Manual.'

— Dilbert

Anweisung Manuell



Wir möchten Sie für das Öffnen dieser wertvollen Zeitschrift beglückwünschen. Indem Sie c't vorwählen, haben Sie eine erstklassige Wahl getroffen. Nur im c't Willen finden Sie immer aktuelle Nachrichten des Stromes ES Fälle, eingehende Berichte der high-end Vorrichtungen und der Anwendungen, ehrliche Produktvergleiche und incisive Analysen der Fälle, die Ihre Zukunft formen.

Kurz gesagt haben Sie die vollkommene Wahl getroffen und wir begrüßen Sie für so Vorwärts-schauen.

Spezifikationen

Dieses Handbuch bedeckt die folgenden Produkte:

- ☞ c't Zeitschrift, Kioskkopie
- ☞ c't Zeitschrift, alleinstehende Version (geborgt)
- ☞ c't Zeitschrift, alleinstehende Version (gekauft)
- ☞ c't Zeitschrift, Subskription Version

Etwas von der Funktionalität, die unten beschrieben ist, ist nicht auf alle Varianten unseres Produktes anwendbar. Etwas Funktionalität kann möglicherweise nicht in etwas geographischen Bereichen vorhanden sein.

Vorkehrungen

Wenn sie für richtig interessiert wird, versieht c't Zeitschrift Sie mit langen Stunden des dauerhaften Genusses. Um den maximalen Genuß aus diesem Produkt heraus zu erhalten, seien Sie bitte sicher unserem nützlichen Führer zu folgen.

- ☞ Wenn es unsachgemäß verwendet wird, kann es unmöglich werden, diese Zeitschrift zu öffnen oder zu lesen.
- ☞ Beachten Sie bitte besonders, dass c't nicht wasserdicht noch feuerfest ist.
- ☞ Halten Sie c't aus der Reichweite der kleinen Kinder, der reizbaren Frauen und der Leute mit einer unausgeglichene Einteilung heraus.

- ☞ Zerreißen Sie nicht heraus Seiten heftig, selbst wenn ihr Inhalt Sie umkippt.
- ☞ Teilen Sie Ihr c't nicht mit Leuten des zweifelhaften moralischen Buchstabens.

Behandlung

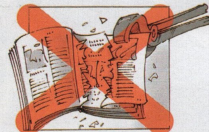
Es gibt einige Weisen, c't Zeitschrift zu benutzen: stationärer und beweglicher Gebrauch.

Wenn Sie planen, c't in einem stationären Klima zu benutzen:

- ☞ Rütteln Sie es, bis alle Blättchen herausgefallen haben.
- ☞ Lesen Sie bitte alle Blättchen, bevor Sie sie sich entledigen; sie können nützliches Informationen und Förderer enthalten anhaltendes Bestehen der c'ts.
- ☞ Spritzenwasser oder andere Substanzen von der Toilette können die Seiten zum Stock zusammen verursachen.
- ☞ Wenn Ihr c't schmutzig erhält, versuchen Sie nicht, die Zeitschrift mit Reinigungsmitteln zu waschen.

Wenn Sie beschließen, diese Zeitschrift unterwegs zu benutzen:

- ☞ beachten Sie bitte, dass das Gehen oder das Fahren beim Ablesen von von c't eingesetzte beide Ihr Leben in Gefahr und Ihre Kopie der Zeitschrift zu beschädigen können.
- ☞ Es ist Ihre Verantwortlichkeit gibt Sorgfalt zur Zeitschrift, um Ihren Genuß zu maximieren (sehen Sie Abschnitt "Vorkehrungen").



Beachten Sie bitte, dass einige Formen der Maximierung der Verwendungsfähigkeit von c't in etwas Bereichen ungültig sein können. Seien Sie sicher, mit Ihren lokalen Behörden zu überprüfen, bevor Sie in gewissem Sinne c't verwenden, das möglicherweise nicht in Ihrem Land erlaubt werden kann. Heiße das Veröffentlichen ist nicht für irgendwelche Beschädigungen verantwortlich, die vom unsachgemäßen oder ungültigen Gebrauch der c't Zeitschrift genommen werden.

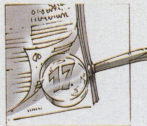
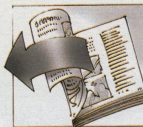
Erster Gebrauch

Überprüfen Sie, ob Sie das Anweisung Handbuch gelesen und verstanden haben, bevor Sie versuchten, irgendwie weiter fortzufahren.

Wenn Sie diese Kopie von c't an einem Zeitung Kiosk grasen, kaufen Sie ihn bitte, bevor Sie fortfahren. Wenn Sie diese Zeitschrift geborgt haben, erwägen Sie, eine Kopie oder eine Subskription zu kaufen.

Nachdem Sie diese Anweisungen durchgeführt haben, fahren Sie zum Index fort, der auf Seiten 6 und 7 gelegen ist.

1. Grasen Sie die Liste der Artikel bis das Finden eines Themas des Interesses zu Ihnen.
2. Folgen Sie den Seitenzahlen, die auf der äußeren Unterseite jeder Seite gedruckt werden, die mit redaktionellem Inhalt gefüllt wird.
3. Wahlweise freigestellt: Überspringen Sie zur gelegentlich lustigen Karikatur auf Seite 17.



4. Nachdem Sie wahlweise freigestellte Abschnitte durchgeführt haben, gehen Sie zurück zu Index.
5. Wenn keines der Themen Ihre sofortige Phantasie anspricht, fangen Sie an, die Zeitschrift von Seite 18 vorwärts zu lesen und fahren Sie fort, bis Sie die letzte Seite erreichen.

Ablezen eines Artikels

Auf der Indexseite wählen Sie einen Artikel des Interesses zu Ihnen vor und merken Sie sich die Zahl, die nahe bei der Einlinie Zusammenfassung des Artikels verzeichnet wird.

- ☞ Schlagen Sie durch die Zeitschrift leicht, bis Sie die Seite erreichen, in der die Zahl auf dem äußeren unteren Rand die Zahl zusammenbringt, die, Sie sich früh merken.

Wenn Sie die Seite gefunden haben, in der der Artikel anfängt, Anfang durch das Ablezen der Schlagzeile, des Untertitels und des einleitenden Punkts.

- ☞ Der einleitende Punkt wird im fetten Gesicht gedruckt.
- ☞ Fahren Sie, indem Sie die sidebars lesen fort, fahren Sie dann zur Zusammenfassung des Artikels fort.
- ☞ Nehmen Sie einen langen Blick an den Tischen, die den Artikel schmücken. Jemand arbeitete sehr stark, um alle Zahlen recht zu erhalten.
- ☞ Wenn Sie folglich geneigt sind, können Sie den Körper des Artikels jetzt lesen.

Wenn Sie Schwierigkeiten einen Artikel lesend oder verstehend haben:

- ☞ treten Sie mit unserem LeserFernsprechdienst bei 511 535 2333 in Verbindung (Montag zu Freitag, 1. P.M. bis 2 P.M.)
- ☞ Haben Sie bitte Ihre bereite Subskription Zahl.
- ☞ Sie können eine E-mail zu <ct@ctmagazin.de> auch schicken.

Technische Unterlagen: G. Himmelein
Gestaltungsarbeit: H.-J. Marhenke

Was ist Benutzerdokumentation?

Definition (Benutzerdokumentation)

ist eine Methode, technische Information einer Software an Nichttechniker zu vermitteln, um sie in ihrer Aufgabe zu unterstützen.

Ist eine Methode... zu vermitteln...

Es ist nur eine Methode unter vielen:

- Schulungen
- Seminare
- Coaching
- Versuch-und-Irrtum
- Selbsterklärungsfähigkeit der Software
- Hotline (Help-Desk)

... Nichttechnikern ...

Benutzer sind keine Software-Experten und wollen auch keine werden.

Sie wollen das wissen, was Sie für Ihre Aufgabe benötigen und nicht mehr.

...um sie in ihrer Aufgabe zu unterstützen...

Ihre Aufgabe ist es nicht, Software zu bedienen, sondern ein reales Problem zu lösen.