

# BBST Bug Advocacy

Session: Altom\_October2016

Student Name: Camelia Chisalita-Cretu

Submitted: October 26, 2016

## Bug Replication

Issue 120368 - font size with decimal values don't have a consistent approximation

## Contents

List of Tables .....	2
List of Figures .....	2
1. Introduction.....	3
2. Bug Report Timeline .....	3
3. Settings .....	3
4. RIMGEN Approach.....	4
4.1. Replicate.....	4
4.1.1. Assess Bug Severity .....	5
4.1.2. Bug Generality.....	11
4.1.3. Assess Product Context .....	12
4.2. Isolate .....	12
4.3. Maximize .....	12
4.4. Generalize .....	12
4.5. Externalize.....	13
4.6. Neutral tone .....	13

## List of Tables

Table 1 Bug Replication for the Font Size value approximation (previous tests)	5
Table 2 Bug replication by varying the behaviour	6
Table 3 Approximation inconsistencies for different features that use Font Size values	8
Table 4 Bug replication by varying the options and settings	9
Table 5 Bug replication by varying the data (Romanian Language was set)	10

## List of Figures

Figure 1 Inconsistency in Font Size values for different application points	8
Figure 2 Inconsistency on Cursor Size when changing Font Size value	8
Figure 3 Cannot reverse to the default dictionary	10
Figure 4 MS Office Word Font Size handling solution for non-numerical input values	11

## 1. Introduction

This is a replication report on the **Issue 120368 - font size with decimal values don't have a consistent** ([https://bz.apache.org/ooo/show\\_bug.cgi?id=120368](https://bz.apache.org/ooo/show_bug.cgi?id=120368)) logged on the Open Office bug tracking system.

I intend to investigate this issue by addressing key aspects on bug reporting writing, suggested by RIMGEN acronym. It corresponds to the following bug research activities: replicate, isolate, maximize, generalize, externalize, neutral tone. Each of them will be approached according to its applicability degree to the investigated issue.

Section 2 shortly reminds the timeline for the studied issue. Technical aspects of the follow-up testing are presented in Section 3 . Section 4 approaches in detail the replication of this issue in various ways, addressing its *severity*, *generality*, and the *product context*.

## 2. Bug Report Timeline

**Issue 120368 - font size with decimal values don't have a consistent approximation** ([https://bz.apache.org/ooo/show\\_bug.cgi?id=120368](https://bz.apache.org/ooo/show_bug.cgi?id=120368)) was:

- first Reported: 2012-07-25 by Mohammed Kuranga;
- last Modified: 2014-10-08 by Heli Desai.

## 3. Settings

**Issue 120368 - font size with decimal values don't have a consistent approximation** will be investigated on the following software and hardware configuration:

- Operating System: Windows 10 Home, 64-bit;
- Processor: Intel Core i5, 1.7GHz;
- RAM Memory: 4Gb;
- Open Office 4.1.3.

## 4. RIMGEN Approach

RIMGEN stands for **R**eplicate, **I**solate, **M**aximize, **G**eneralize, **E**xternalize, **N**ormal tone. RIMGEN is a recommended approach in writing bug reports.

### 4.1. Replicate

The follow-up testing will achieve several types of investigation:

- assess the **bug severity** with the 4 "vary"-ies:
  - vary the behaviour;
  - vary the options and settings;
  - vary data files;
  - vary the configuration;
- assess the **bug generality**:
  - failure conditions
    - extreme-value tests;
    - corner case tests;
  - configuration dependence;
  - different ways lead to this bug;
  - bug age;
    - the bug is new to this version;
    - similar failures appear in the database;
  - similar bugs appear in other programs;
- assess the **context**:
  - competitor;
  - product itself;
  - product owner;
  - end user/client/consumer.

### 4.1.1. Assess Bug Severity

In order to replicate the issue the following scenario template (**ScenarioA**) consisting of 6 steps was applied:

1. Start Apache OpenOffice 4.1.3;
2. Create a new Text Document;
3. *[optional]* Enter text, e.g., "abc"; initial Font Size is set to 12;
4. *[optional]* Select the entered text;
5. In the main menu, in the Font Size dropdown list enter a **new Font Size value**, e.g., 14.23;
6. Press **Enter** to update **Font Size value** (updated to 14.1 for the above input);

**Expected result:** Font Size value is updated according to the Specification Document or an error message should appear when attempting to insert a non-numeric value;

**Actual result:** Font Size value is updated as:

- for **input values with decimals** the approximation is inconsistently achieved (i.e., it does not follow the mathematical rounding);
- for **input values with no decimals** the *output value* equals to the *input value*;

Table 1 shows several Font Size values approximation test cases replicated from the **initial bug report**;

TC No.	Font Size value			
	Existent (Current)	Entered (Input)	Result (Actual Output)	Resulted (Expected Output)
TC01	12	3000	999.9	<error message> or 12
TC02	12	100000	999.9	<error message> or 12
TC03	12	1000	999.9	<error message> or 12
TC04	12	999.99	999.9	<error message> or 12
TC05	12	999.1	999	999.1
TC06	12	2.2	2.2	2.2
TC07	12	2.3	2.2	2.3
TC08	12	2.4	2.4	2.4
TC09	12	2.5	2.5	2.5
TC10	12	2.6	2.5	2.6
TC11	12	2.7	2.7	2.7
TC12	12	2.8	2.7	2.8
TC13	12	100.6	100.5	100.6
TC14	12	100.7	100.6	100.6

Table 1 Bug Replication for the Font Size value approximation (previous tests)

## Bug severity

The bug severity was studied investigating the Writer's behaviour where changes or variation on features run, data files, settings, or configuration occurs.

### a. Vary the behaviour

- Scenarios:

1. change the Font Size value from:
  - i. the Font menu (Format-> Character) ([TC15]);
  - ii. the main menu (Font size drop down list) ([TC16]);
  - iii. the right click Size menu (not allowed, predefined values to select only);
2. change the Font Size value for text in:
  - i. document ([TC16]);
  - ii. tables ([TC17]);
  - iii. footnote content ([18]);
3. change the Font Size value with a not valid value:
  - i. document ([TC19], [TC23], [TC24], [TC25]);
  - ii. the Font menu (Format-> Character) ([TC20]);
  - iii. tables ([TC21]);
  - iv. footnote content ([TC22]);

TC No.	Font Size value			
	Existent (Current)	Entered (Input)	Result (Actual Output)	Resulted (Expected Output)
TC15	88.5	99.23	99.1	99.2
TC16	17	4.56	4.5	4.6
TC17	29	1209	999.9	<error message> or 29
TC18	12	200.4	200.3	200.4
TC19	45	4523.	999.9	< error message> or 45
TC20	12	abcde	12	12
TC21	56.9	13-g	13	< error message> or 56.9
TC22	766	M-43.9	43.9	< error message> or 766
TC23	36	3e3r	33	< error message> or 36
TC24	33	3e3r3t	333	< error message> or 33
TC25	33	4r5t6y7u8i	999.9	< error message> or 33
TC26	45.5	78.23	78.1	78.2
TC27	45.5	78,23	999.9	< error message> or 45.5
TC28	45.5	4,5	45	< error message> or 45.5

Table 2 Bug replication by varying the behaviour

### Other inconsistencies found during testing

For [TC26], after changing the Font Size value to 78.23, the approximated value is 78.1. This is the first failure. ScenarioA was run here.

**ScenarioB** is a variation of ScenarioA, considering the Font menu window to update the Font Size value. ScenarioB consists of the following steps:

1. *[optional]* Exit OpenOffice Writer 4.1.3 then start again OpenOffice Writer 4.1.3;
2. *[optional]* Create a new Text Document;
3. Choose **Format** menu;
4. Choose **Character** submenu; the **Font** configuration windows is shown;
5. Type in Font Size dropdown list a **new Font Size value**, e.g., 14.23;
6. Press **Enter** to update **Font Size value**;

- a. the Font window is closed now and in the main menu, in the Font Size dropdown list the value is updated to 14.1 (for the above input);
- 7. Choose **Format** menu again;
- 8. Choose **Character** submenu;
  - a. the **Font** windows is shown where the Font Size value may be different (14.2 for the above input);

**Expected result:** Font Size value from the main menu should be updated according to the Specification Document or an error message should appear when attempting to insert a non-numeric value;

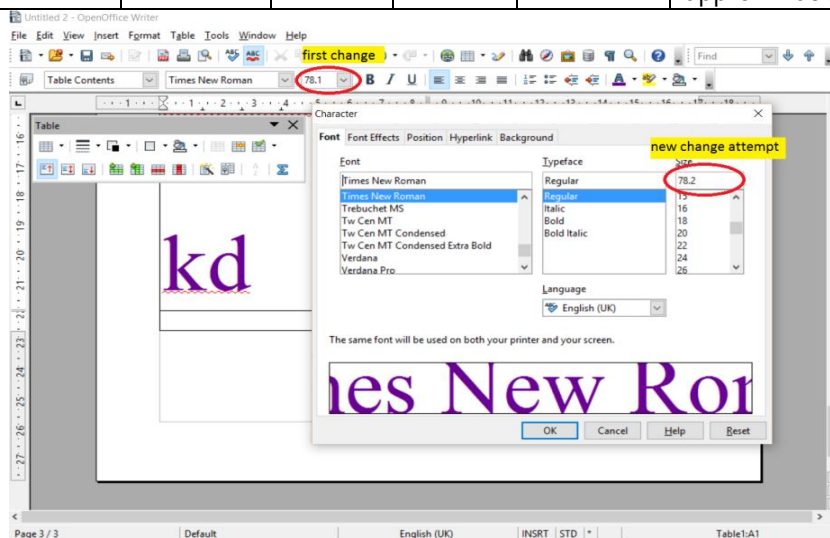
**Actual result:** Font Size value is updated as:

- for **input values with decimals** the approximation is inconsistently achieved (i.e., it does not follow the mathematical rounding);
- for **input values with no decimals** the *output value* equals to the *input value*;

ScenarioB was run considering the input value 78.23, again.

Figure 1 captures the program state where Font Size value on step 8 in ScenarioB, where the behaviour is inconsistent. The font Size attribute has assigned two different values. This is the second failure.

Execution order	Current value	First Scenario		Second Scenario		Notes
		Input	Output	Shown Value	Updated?	
1.A, 2.B	100	78.23	78.1	78.2	yes	updated value, correct approximation in B, wrong approximation in A
1.B, 2.A	100	78.23	78.2	78.1	yes	updated value, wrong approximation in A
1.A, 2.B	78.2	100	100	100	yes	updated value, correct approximation
1.B, 2.A	78.2	100	100	100	yes	updated value, correct approximation



Execution	Current	First Scenario	Second Scenario	Notes
-----------	---------	----------------	-----------------	-------

		Input	Output	Shown Value	Updated?	
1.A, 2.B	100	78.23	78.1	78.2	yes	updated value, correct approximation in B, wrong approximation in A
1.B, 2.A	100	78.23	78.2	78.1	yes	updated value, wrong approximation in A
1.A, 2.B	78.2	100	100	100	yes	updated value, correct approximation
1.B, 2.A	78.2	100	100	100	yes	updated value, correct approximation

Figure 1 Inconsistency in Font Size values for different application points

The inconsistency persists when the ScenarioB is run before ScenarioA. Table 3 shows several runs where ScenarioA and ScenarioB are executed in different order with different input.

Table 3 Approximation inconsistencies for different features that use Font Size values

### Considerations on this failure

Both ScenarioA and ScenarioB work on the same input. The proof is the values are both updated. But they are updated with different values (in case input has decimals). The different value indicates that:

- The result value is computed differently (it may be a different approximation algorithm);
- The dropdown list from the main menu and the Font menu window do not collaborate in updating values one from the other.

### Other results of the follow-up testing

Figure 2 shows another inconsistency when changing the Font Size value, e.g., from 12 to 100. The Cursor Size is kept unchanged.

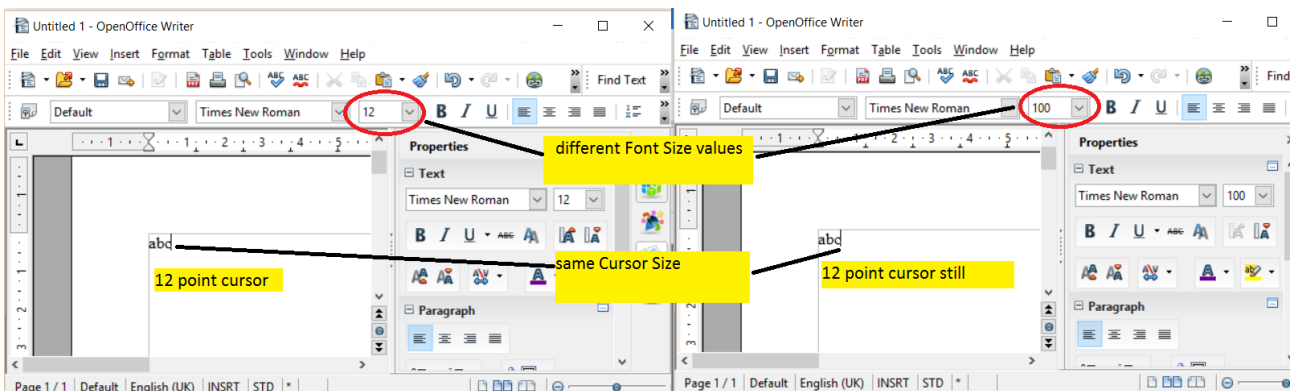


Figure 2 Inconsistency on Cursor Size when changing Font Size value

#### b. Vary the options and settings;

- **Scenarios:**
  1. change the measurement unit (Options -> OpenOffice Writer -> General -> Measurement Unit):
    - i. from **centimetres** to **inches**;
    - ii. from **inches** to **point**;
  2. change the view (View):



- i. from **Print Layout** to **Web Layout**;
- ii. from **Web Layout** to **Print Layout**;

Test cases TC15 to TC28 were run for combined scenarios 1.i and 2.i. No changes on the test results.

TC No.	Font Size value			
	Existent (Current)	Entered (Input)	Result (Actual Output)	Resulted (Expected Output)
TC15	88.5	99.23	99.1	99.2
TC16	17	4.56	4.5	4.6
TC17	29	1209	999.9	<error message> or 29
TC18	12	200.4	200.3	200.4
TC19	45	4523.	999.9	<error message> or 45
TC20	12	abcde	12	12
TC21	56.9	13-g	13	<error message> or 56.9
TC22	766	M-43.9	43.9	< error message> or 766
TC23	36	3e3r	33	< error message> or 36
TC24	33	3e3r3t	333	< error message> or 33
TC25	33	4r5t6y7u8i	999.9	< error message> or 33
TC26	45.5	78.23	78.1	78.2
TC27	45.5	78,23	999.9	< error message> or 45.5
TC28	45.5	4,5	45	< error message> or 45.5

Table 4 Bug replication by varying the options and settings

**c. Vary data files;**

**- Scenarios:**

1. change the dictionary for the spelling options (Tools -> Spelling and Grammar- > Add *standard.dic*);

I have run some of the test cases from TC15 to TC28. No changes on the test results.

2. change the language for all text to Romanian (Tools -> Language - > For all Text -> More... -> Languages -> Local Setting -> choose *Romanian*); start again OO Writer;

Test cases TC15 to TC28 were run again. Table 5 shows the results. By varying the data (i.e., in this case the Language to Romanian, decimal point . is changed to ,). For this type of variation test cases have different identification numbers (e.g., TC15 for English language settings and TC27 for Romanian language settings). The expected result is different. There a new column that states the expected result when for other types of variation that used decimal point (.). the result shows consistency with the language settings.

TC No.	Font Size value						
	Existent (Current)	Entered (Input)	Result (Actual Output)	Resulted (Expected Output)	Resulted (Expected Output) with <b>decimal point</b>		
TC27	88,5	99.23	999,9	<message> or 88,5	TC15	99.1	99.2
TC28	17	4.56	456	<message> or 17	TC16	4.5	4.6
TC29	29	1209	999,9	<message> or 29	TC17	999.9	<message> or 29
TC30	12	200.4	999,9	<message> or 29	TC18	200.3	200.4
TC31	45	4523.	999,9	<message> or 45	TC19	999.9	<message> or 45
TC32	12	abcde	12	<message> or 12	TC20	12	<message> or 12
TC33	56.9	13-g	13	<message> or 56,9	TC21	13	<message> or 56.9

TC34	766	M-43.9	439	<message> or 766	TC22	43.9	<message> or 766
TC35	36	3e3r	33	<message> or 36	TC23	33	<message> or 36
TC36	33	3e3r3t	333	<message> or 33	TC24	333	<message> or 33
TC37	33	4r5t6y7u8i	999,9	<message> or 33	TC25	999.9	<message> or 33
TC38	45,5	78.23	999,9	<message> or 45,5	TC26	78.1	78.2
TC39	45,5	78,23	78,1	78,2	TC27	999.9	<message> or 45.5
TC40	45,5	4,5	4,5	4,5	TC28	45	<message> or 45.5
TC41	766	M-43,9	43,9	<message> or 766	N/A		
TC42	12	200,4	200,3	200,4	N/A		

Table 5 Bug replication by varying the data (Romanian Language was set)

### Other results of the follow-up testing

OO Writer exposes awkward behaviour though, i.e., it does not allow reversing to the default dictionary. Figure 3 shows the message about completing the spellcheck then it does not allow to change the dictionary and the window automatically closes.

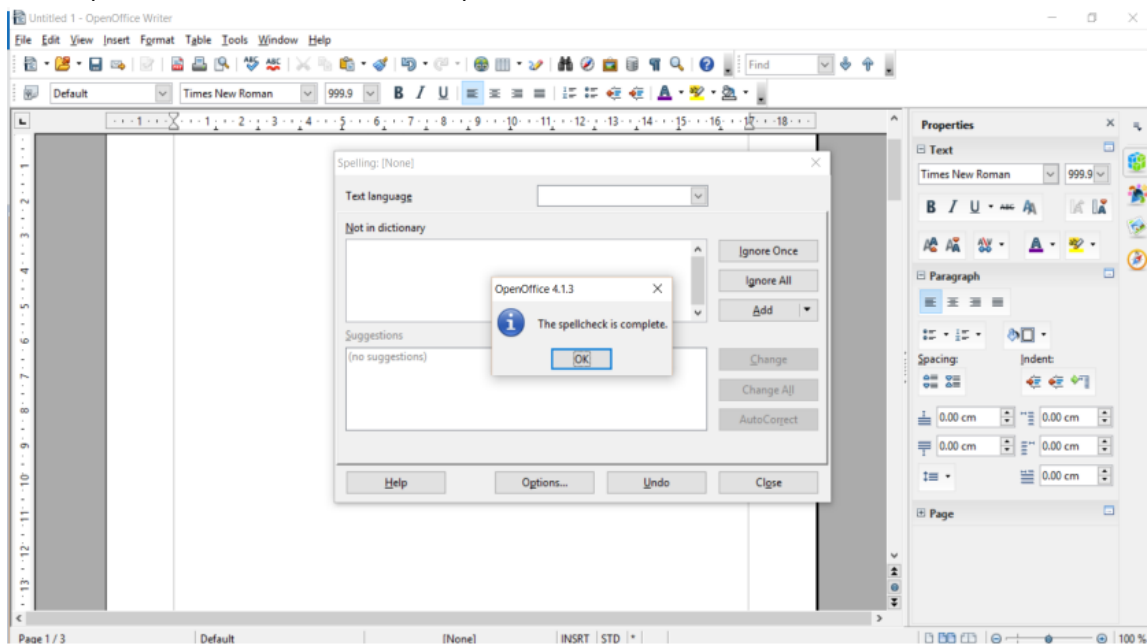


Figure 3 Cannot reverse to the default dictionary

#### d. Vary the configuration;

##### 1.1. Scenarios:

- A. change the *power settings* for the computer:
  - i. from **high performance** to **power saver**;
  - ii. from **power saver** to **high performance**;

Some of the Test cases TC16 to TC28 were run again. There no changes in results. It took OO Writer more time to start after the power settings were set to **power saver**.

## 4.1.2. Bug Generality

In order to assess the **bug generality** the following research was accomplished:

### 1. Analysis on failure conditions indicated by corner case tests

Test cases TCo1 to TCo4 represents corner cases. Further investigation with less extreme values showed that the failure persists. Therefore, test cases TCo6 to TC28 indicate the same failure when Font Size attribute was set to a much smaller value in the mainstream.

### 2. Analysis on configuration dependency

The initial bug report indicated Mac OS X 10.7 the operating system where this bug has shown. Still the bug is replicated on other versions of Mac OS and Windows 7 operating systems. As stated in the Settings section, Windows 10 operating system was used here to replicate this bug. It can be concluded that this is not a configuration dependent failure.

### 3. Analysis on different paths that lead to the same failure

Table 3 shows there is no sequence dependency on showing the failure.

### 4. Analysis on the bug's age:

The bug is reported in OpenOffice 3.4.0. This bug was replicated [here](#) for version OpenOffice 4.1.3. The search on the Bugzilla database didn't show similar bugs on Font Size attribute to be already reported or similar failures on approximation.

### 5. Analysis on similar bugs to appear in other programs:

This failure was investigated on the competitor's product (MS Office Word). This failure didn't show up. The various types of entries (i.e., valid or non-valid, numerical or not) are handled by the competitor's product very strictly:

- If the value is not numerical then an error message is shown (see Figure 4);
- If the value needs approximation to 0.5 then an error message is shown;
- If the value exhibits and corner case then an error message is shown;

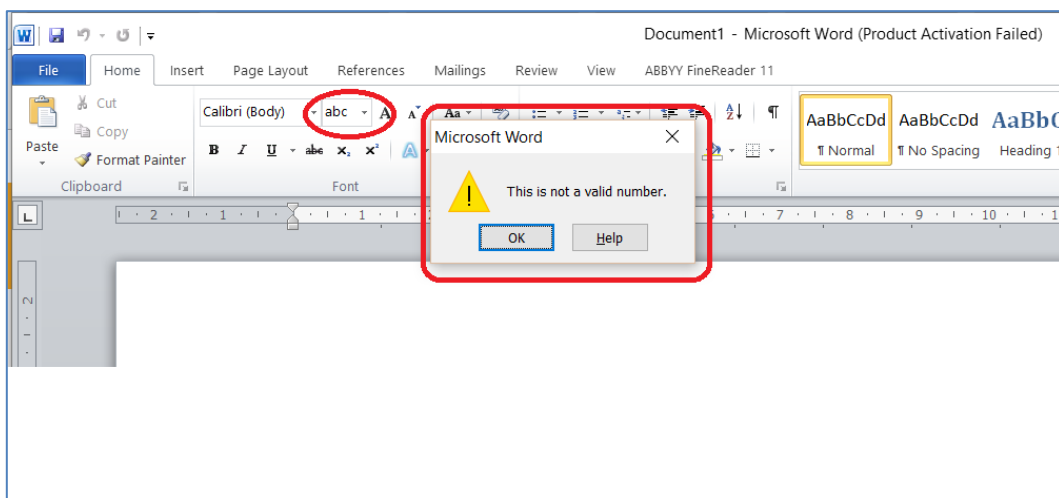


Figure 4 MS Office Word Font Size handling solution for non-numerical input values

### 4.1.3. Assess Product Context

#### 1. Analysis on the competitor failure and provided solution or the way it was avoided:

The competitor (MS Office) avoided the occurrence of this failure by considering a small range of values to be valid for the Font Size. Therefore, any value that does not follow the considered rules is rejected and the value is reversed to the previous one.

OO Writer is very flexible on allowing entries to be considered Font Size values. Test cases TC20 to TC40 indicate a plethora of situations where entry values may be interpreted.

#### 2. Analysis on the product itself:

The impact on the product itself and its capability to satisfy users' need is not affected drastically.

#### 3. Analysis on the product owner:

The owner always tries to keep a high quality standard to the product. Therefore, if the failure proves to affect the needs of many users in a critical manner or too often it will be eventually fixed.

#### 4. Analysis on end user/client/consumer needs:

The product is the second professional word processor tool on the market. Being acquired without any costs the user accepts the inherent failures. He conveys to the situation and tries to avoid the cases where this failure may occur.

## 4.2. Isolate

Test cases presented in Table 3 indicate an easy way to isolate the bug. ScenarioA and ScenarioB allow to compute an approximation of the Font Size value. They provide a different value which means that (at least) one is faulty. There may be used an Oracle that helps to decide the correct result. In this case, ScenarioB returns the right value, while ScenarioA constantly provides a wrong approximation. Therefore, we have:

- ScenarioA that shows the failure of approximation;
- ScenarioB that helps to build various scenarios to isolate the bug in ScenarioA, but it proves the result is not passed to the main window menu.

ScenarioA and ScenarioB transmit one to another the value to be approximated but each of them seem to use a different approximation strategies (i.e., algorithms).

## 4.3. Maximize

**Maximize the worst impact.**

The worst impact is provided by the inconsistency on the two discussed scenarios. The error in approximation is considered less important than the fact Font Size value may be assigned to different values at the same time.

## 4.4. Generalize

**All the platforms have the same bug all**

The failure indicated in the initial report seems to be a small one, an approximation error that affects few people. Few people use the Font Size values having decimals.

#### **4.5. Externalize**

The bug impact may be analysed under different aspects of the end use/client perspective:

- once the client identifies a problem with approximation, he will be very suspicious on any measurements the product needs to work on or to approximate;
- if he realizes that ScenarioB offer him the good approximation result will rely on it and his user experience will be affected; he will have to keep in mind always to avoid ScenarioA instead of ScenarioB.

#### **4.6. Neutral tone**