

From Jeremy Thorne:

It has come to my attention that a number of users of the new EclipseVox are experiencing difficulties with numbers. I'd like to address that issue generally.

First, be aware that the EclipseVox now uses the raw recognition results from Dragon, meaning that whatever you say gets transmitted directly to Eclipse for immediate use before Dragon's own number formatting procedures have a chance to work with it.

For example: Where before you might say (A)"It cost three thousand five hundred dollars for that item" and would get (B)"It cost \$3,500 for that item" from Dragon, what Eclipse will NOW see is the original text (A).

Fortunately, Eclipse contains number formatting that is much more customizable and sophisticated than Dragon's. That customizability, however, means that it's possible to set it up so that numbers don't appear the way you want them to.

Here are some recommendations for dictionary entries and number settings for EclipseVox:

Under User settings/Numbers:

Number conversion: ON (otherwise, you'll get "three thousand five hundred", etc.)

Ignore written numbers: OFF (the word "three" should be recognized as a number in the above example. If you ignore written numbers, they'll never be converted.)

Process "and" as a number: OPTIONAL (If you say "and" inside a number such as "five hundred and seven" or "three dollars and ten cents" then turn it on. If you can remember NOT to say "and" then you should have it turned off.)

Force measurements to digits: OPTIONAL (Should "three miles" always be "3 miles?" If so, turn it on.)

Force separators in currency: OPTIONAL (Should "twenty-five hundred dollars" be (ON)\$2,500 or (OFF)\$2500? Note that "two thousand five hundred dollars" will ALWAYS be \$2,500.)

Extra ordinal formatting: OPTIONAL (Should the "rd" in "23rd" be superscripted? If so, turn it ON. Note that this will also force the ordinal suffix to be lower case when doing all-caps captioning or CART.)

Glue numbers: OPTIONAL (ON: "I live in apartment 5B. I worked for the FBI5 years ago." OFF: "I live in apartment five B. I worked for the FBI five years ago." Neither is perfect.)

Minimum quantity size: 0 or 5 (A quantity is any number containing the word "thousand" or greater, such as "million" or "billion." With this value set to 0, such words will always contain commas. With this value set to 5, such words will only contain a comma if they're at least five digits in length. The fact that we're beyond the year 2000 makes this ambiguous. Even though most users prefer that "there were five thousand people at the rally" would read "there were 5,000 people at the rally" most users don't want "it was back in two thousand seven when that happened" to come up "it was back in 2,007 when that happened." Having 2007 may be worth the tradeoff of having 5000 in the former example.)

One other way to deal with the year is to leave the minimum quantity size at 0 and put "two thousand = 2{#G}thousand" in your main dictionary. If you do that, then any time that you speak ANY number containing "two thousand" it will translate as a Generic number rather than a quantity and will never have commas in it. This fixes all years, but if you say "there are fifty two thousand seven hundred containers here" you will get "there are 52700 containers here."

Time: 9:00 a.m. (this setting is the most common for times.) You'll note that the metadictionary already contains entries for a.m. and p.m. that force the previous number to be a time, so you do NOT need a dictionary entry such as {#T}a.m. If you say "nine A M" and you get something like nine AM, you may want to define AM = a.m. in your main dictionary. Some users prefer 9 a.m., and if you do, set the Time setting to 9 a.m. instead.

Digits = : This setting determines the default number type for the number being converted based on the number of digits. This is useful if no other trigger (such as {#T} or {#G}) was written, or if no other indication exists (such as the word "thousand" which would make it a quantity.) By default, 1-digit, 2-digit and 3-digit numbers should be set to Quantity. All others should be set to Generic unless you want it to automatically assume, for example, that ten digits with no triggers should be a phone number. Having 5-digit numbers set to Generic, for example, means that you can say a zip code such as "three four nine five three" and get 34953 instead of 34,953.

Numeric classes: There are several users who need to change these settings. For each class (Quantity, Currency, Ordinal) there are three settings below. The "write out" rule tells when to write out the word instead of using the digit. The "At start" rule is the same as the write out rule, except it only applies at the start of a sentence. The "Clarify" setting tells what minimum size of large number should leave the word if no digits follow. The default setting is "million+" meaning that "fifty two million" would be "52 million" and not "52,000,000".

If you want "I have three dogs" to appear exactly that way and not "I have 3 dogs" then you must make sure that the "write out" value for quantities includes the number 3, such as setting it to "1-10." Most users set it that way. Most users set the "at start" to 1-100. Note that the settings can be different for currency and ordinal numbers, so if you always want \$5 to appear in digits, it can be set to "write out: none" instead of 1-10.

It takes a little while, but you have to get used to the step-by-step concept of how Eclipse processes numbers. If you say "I have three dogs"

let's look at what Eclipse is doing:

Is Number conversion on? YES -- Get ready to convert Is Ignore written numbers on? NO -- The word "three" qualifies as a number with a value of 3 (according to the number vocabulary setup, which is customizable for different languages.) How many digits is the number? -- One What sort of number is it? Since there are no triggers, 1-digit = Quantity, so it's a quantity.

Should it be written out? Well, let's see:

... Is it a Quantity? YES

... Is it at the start of a sentence? NO ... Is it in the 1-10 range indicated by the "write out" setting?

YES So convert it to a written quantity: "three"

Now, that may seem like a lot of processing to be right back where we started, but it's precisely because of the different ways people want numbers to appear in different circumstances that all of that processing is necessary.

One final note:

There is a LOT that can be done with this feature to customize it. For example, the customizable number templates allow you to do things like account numbers such as 34-5746-3637-(36) without having to speak hyphens or parens just by making a template such as ##-####-####-(##). You can also incorporate triggers into dictionary entries to force numbers to convert a certain way in a particular context, such as defining template 1 as .### and then defining "caliber = {#1}caliber" in your main dictionary so you can say "it was a 38 caliber revolver" and get "it was a .38 caliber revolver."

It's very hard to come up with a number conversion example that Eclipse can't handle one way or another. Once you have gone through and customized the settings to fit your preferences, if you are still having problems, let me know and I'll see if I can get you some step-by-step instructions about how to address the issues.

Jeremy

P.S. Some voice users have created conflicts such as to = \to\two and two = \two\to. If Dragon is generally getting the right word, it may be advisable not to use these conflicts, because once a word has been included in a number conversion, it becomes part of a larger entity and can no longer be selected as a conflict.