OPENRULES®

Open Source Business
Decision Management System
Release 6.1.3

Preparing a Tax Return Using Executable Decisions (Decision1040EZ)

Tutorial

OpenRules, Inc.

www.openrules.com

December-2011

Table of Content

Introduction

In this tutorial we explain how to implement a rules-based application that calculates tax returns using the notorious US tax form 1040EZ. This tutorial is written in the form of a dialog between two people:

- User – a person who wants to learn how to write OpenRules®-based applications
- OR – a representative of OpenRules® technical support who explains how to do it.

A user is not expected to be a software developer but rather a business analyst who is familiar with Excel and who has already looked at the OpenRules document "Getting Started" that includes several basic examples.

This tutorial explains how to use OpenRules® Decisioning approach to implement tax calculation logic for the form 1040EZ. It does not include any GUI and can be incorporated in any application as a pure decision service.

There is also another tutorial that explains how to use OpenRules® Dialog (ORD) to create a web-based dialog during which only necessary questions are being asked, all other answers are automatically calculated, and a ready to go tax return in the PDF format is being produced – see http://openrules.com/pdf/Tutorial.Dialog1040EZ.pdf.

Business Case

OR. We will prepare the notorious US tax form 1040EZ used by many US taxpayers to report on their revenue and expenses and calculate whether the taxpayer owes additional money or can expect a refund.

User. I did not use it myself but I know how frustrated some of my colleagues were when they tried to fill out this form themselves.

OR. There are plenty of web-based applications that help people to do it. While the form 1040EZ is supposed to be used only in simple cases, the business logic behind this form is not so simple, especially when a taxpayer was shown as a dependent by his/her parents or somebody else. The actual form has only 2 pages and in many cases only the first page is needed. The second page has to be filled only when a taxpayer is shown as a dependent. We also have to keep in mind the eligibility criteria that specify when to use this form.

User. Where can we get the form?

OR. Look at the Appendix below. We downloaded this form from the IRS website and used this form to demonstrate how to use OpenRules®. For your convenience, the form 1040EZ is presented on the next two pages. We use the tax form for 2003 as the example for our application – see the Appendix below. You may download the latest 1040EZ form from http://www.irs.gov/pub/irs-pdf/f1040ez.pdf. The instructions can be found at http://www.irs.gov/pub/irs-pdf/i1040ez.pdf. As you can see, the form is self-explanatory and people should be able to fill it out without external help.

User. I noticed that the official 1040EZ instructions have more than 35 pages; probably it is not so easy to cover all possible situations. However, our real OpenRules®-based application is also not going to be easy, so I'd appreciate it if you would explain how to handle not just simple cases but also much more complex business scenarios.

OR. OK, we plan to build a simple decision service Decision1040EZ that implements only the business logic of the form 1040EZ.

Decision "Decision1040EZ"

OR. This section will describe a methodological approach and consider different implementation options for the above tax form 1040EZ. A ready to go application is a part of the OpenRules® installation – you can find it in the sample project "Decision1040EZ".

Starting with Decision

OR. We will apply a top-down development approach. It means that we will start with the definition of a Decision and not with rules or data. Let's call our decision "DetermineTaxReturn".

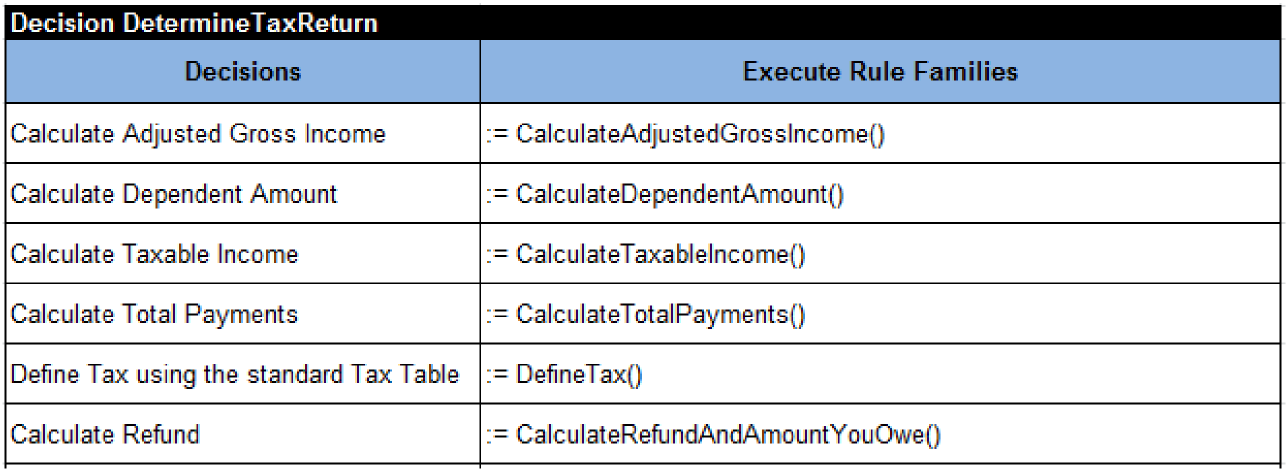
User. Sounds good to me.

OR. Let's look at the first page of the Form 1040EZ. Can you list all decisions that we should make?

User. I guess they are defined in the lines of the form. After filling out general information about a taxpayer we should do the following calculations:

- Calculate Adjusted Gross Income (Line 4 by adding the content of Lines 1, 2, and 3)
- Calculate Dependent Amount (Line 5 with potentially complex logic described on the second page)
- Calculate Taxable Income (Line 6)
- Calculate Total Payments (Line 9)
- Define Tax (Line 10 using the standard Tax Table)
- Calculate Refund or Amount You Owe

OR. Good. So, let's create an Excel file "Decision.xls" with the following table of the type "Decision":



Decision DetermineTaxReturn	
Decisions	**Execute Rule Families**
Calculate Adjusted Gross Income	:= CalculateAdjustedGrossIncome()
Calculate Dependent Amount	:= CalculateDependentAmount()
Calculate Taxable Income	:= CalculateTaxableIncome()
Calculate Total Payments	:= CalculateTotalPayments()
Define Tax using the standard Tax Table	:= DefineTax()
Calculate Refund	:= CalculateRefundAndAmountYouOwe()

The first column of this table defines the exact names for all of the above decisions. We will need to create decision tables for each of these decisions but at this point it is enough to just give these decision tables exact (unique) names – see the second column.

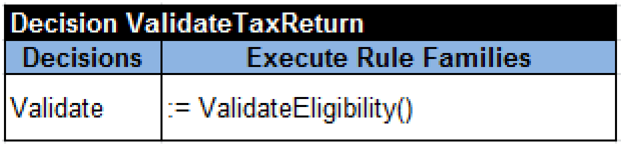
User. Can you remind me why each decision table name starts with `:=` and ends with `()`?

OR. This is necessary to inform OpenRules® that decision tables are actually executable Java functions contrary to the decision names in the first column.

User. Now we can start creating Excel tables for the Excel-based decision tables.

OR. Not so fast. If you look at the second page of the 1040EZ from you will notice some limitations when this form can and cannot be used. It means that before we start any calculations we should validate if the taxpayer is eligible to use this form. I'd recommend defining two different decisions, one that validates a taxpayer eligibility to use this form and another that actually does the necessary 1040EZ calculations.

User. OK, we can add another decision in a separate decision table:



Decision ValidateTaxReturn	
Decisions	**Execute Rule Families**
Validate	:= ValidateEligibility()

But how can we connect these two decisions? We should execute the first decision only when the second one is successful.

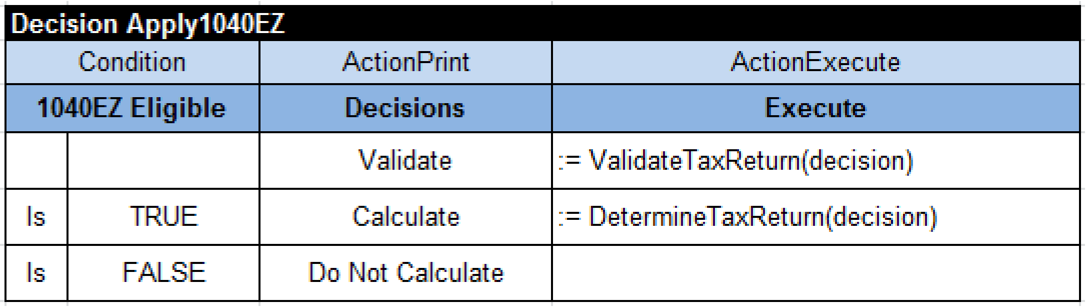
OR. We may define a top-level decision with the following logic:

1. Execute a sub-decision that validates a taxpayer's 1040EZ eligibility
2. If a taxpayer is 1040EZ eligible
   Then execute sub-decision that calculates the tax return.

Let's call this top-level decision "Apply1040EZ".

User. OK, but I did now know that decisions can have sub-decisions.

OR. Actually, you can use multiple tables of the type "Decision" to represent any decision trees just in a tabular format (that frequently is more manageable than a graphical tree). You even can use conditions inside decision tables. Let me show you a possible solution:



Decision Apply1040EZ			
Condition		ActionPrint	ActionExecute
1040EZ Eligible		Decisions	Execute
		Validate	:= ValidateTaxReturn(decision)
Is	TRUE	Calculate	:= DetermineTaxReturn(decision)
Is	FALSE	Do Not Calculate	

Because we use an optional "Condition" we have to add a second row. The keywords "Condition", "ActionPrint", and "ActionExecute" are defined in the standard OpenRules template "DecisionTemplate" – see the configuration file "DecisionTemplates.xls" in the folder "openrules.config". Here I am using a decision variable "1040EZ Eligible" that we will add to our glossary later on (probably as an attribute to a business concept "TaxReturn"). I assume that decision "ValidateTaxReturn" should set this decision variable to TRUE or FALSE.

User. Interesting… And if our validation fails (1040EZ Eligible will be FALSE) you just want to print the words "Do Not Validate" but do nothing.

OR. Exactly right. Note that the decision "ValidateTaxReturn" will always be executed (no conditions are defined). Also note that contrary to the decision tables, decisions always use a predefined parameter `(decision)` as defined in the template.

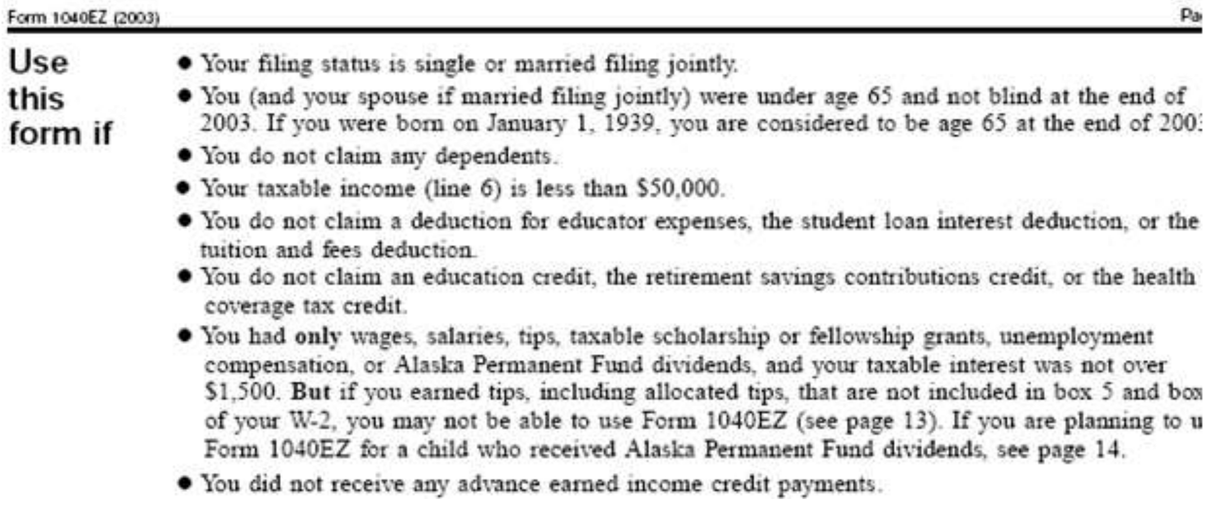
Now, when we are done with the definition of our decision and sub-decisions, we may concentrate on the decision tables.

Defining Decision Tables

OR. The form 1040EZ gives us an opportunity to demonstrate different types of rules starting from very simple validation decision tables, then rules that deal with different calculation formulas, and finally with quite complex rules for calculation of dependent amounts. Let's start with validation rules for the names we already specified in the decision "ValidateTaxReturn".

Validation Rules – Simple Decision Tables

User. OK. Eligibility rules are defined at the beginning of the second page of Form 1040EZ:



Form 1040EZ (2003) Pa

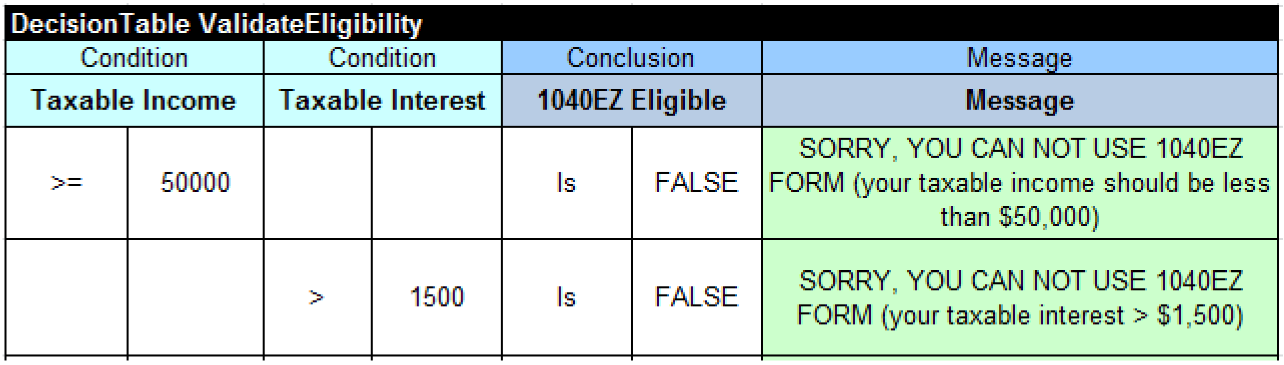
Use this form if

- Your filing status is single or married filing jointly.
- You (and your spouse if married filing jointly) were under age 65 and not blind at the end of 2003. If you were born on January 1, 1939, you are considered to be age 65 at the end of 200
- You do not claim any dependents.
- Your taxable income (line 6) is less than $50,000.
- You do not claim a deduction for educator expenses, the student loan interest deduction, or the tuition and fees deduction.
- You do not claim an education credit, the retirement savings contributions credit, or the health coverage tax credit.
- You had **only** wages, salaries, tips, taxable scholarship or fellowship grants, unemployment compensation, or Alaska Permanent Fund dividends, and your taxable interest was not over $1,500. **But** if you earned tips, including allocated tips, that are not included in box 5 and box of your W-2, you may not be able to use Form 1040EZ (see page 13). If you are planning to u Form 1040EZ for a child who received Alaska Permanent Fund dividends, see page 14.
- You did not receive any advance earned income credit payments.

I think we can automatically validate only two eligibility conditions:

1. Your taxable income (line 6) is less than $50,000
2. You taxable interest was not over $1,500

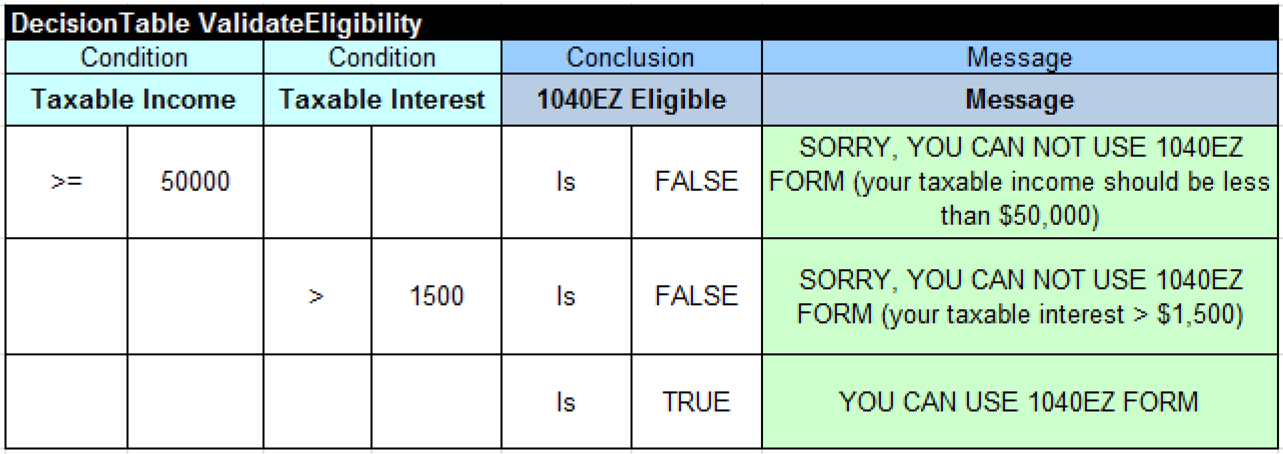
So, I can create the proper decision table "ValidateEligibility" as follows:



DecisionTable ValidateEligibility						
Condition		Condition		Conclusion		Message
Taxable Income		Taxable Interest		1040EZ Eligible		Message
>=	50000			Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable income should be less than $50,000)
		>	1500	Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable interest > $1,500)

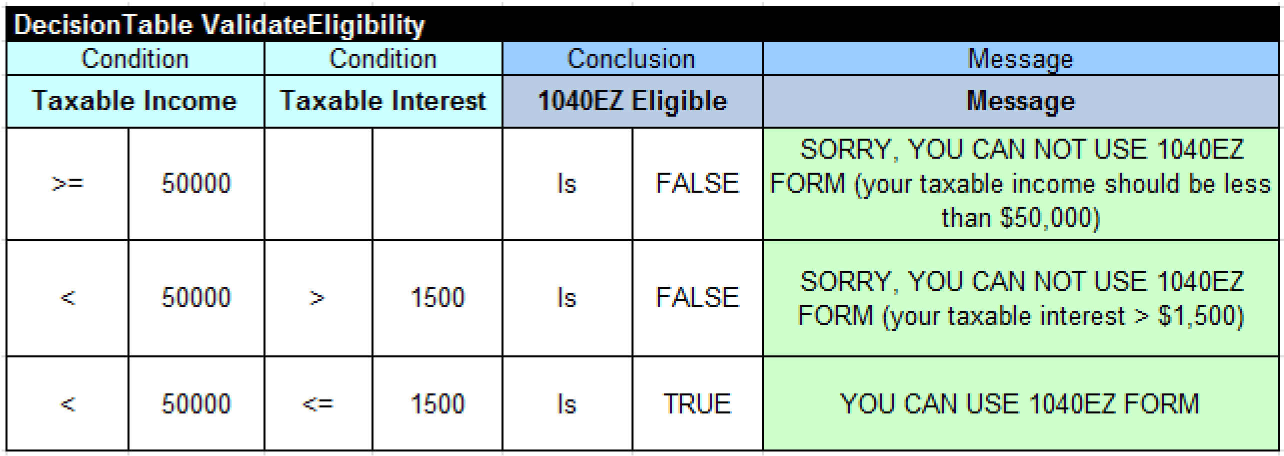
OR. Looks good but I can see two problems here. First of all, what will happen if both eligibility conditions are met? In other words, who will set the decision variable "1040EZ Eligible" to TRUE?

User. It is easy to fix. I will add one more row at the end of the table that will set this decision variable to TRUE and will even produce a message that 1040EZ can be used. Here it goes.



DecisionTable ValidateEligibility						
Condition		Condition		Conclusion		Message
Taxable Income		Taxable Interest		1040EZ Eligible		Message
>=	50000			Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable income should be less than $50,000)
		>	1500	Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable interest > $1,500)
				Is	TRUE	YOU CAN USE 1040EZ FORM

OR. The OpenRules® engine will actually execute this decision table in the way you expect. However, theoretically this decision table violates a requirement that the order of rules (rows) inside a decision table should not matter. In this case it is easy to fix the reliance on the rule ordering by simply avoiding empty conditions as shown below.



DecisionTable ValidateEligibility						
Condition		Condition		Conclusion		Message
Taxable Income		Taxable Interest		1040EZ Eligible		Message
>=	50000			Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable income should be less than $50,000)
<	50000	>	1500	Is	FALSE	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable interest > $1,500)
<	50000	<=	1500	Is	TRUE	YOU CAN USE 1040EZ FORM

User. I still like my initial version better because it is simpler. With your approach we have to mention thresholds like "50000" and "1500" in several places. What if we want to change such thresholds? What if we have not 2 but 4 or 5 condition columns? It would be much more difficult to maintain such a decision table.

OR. Your arguments are well taken. In reality it is really up to a rule designer (or to the company's official policy) to decide how strictly to follow the methodological principles.

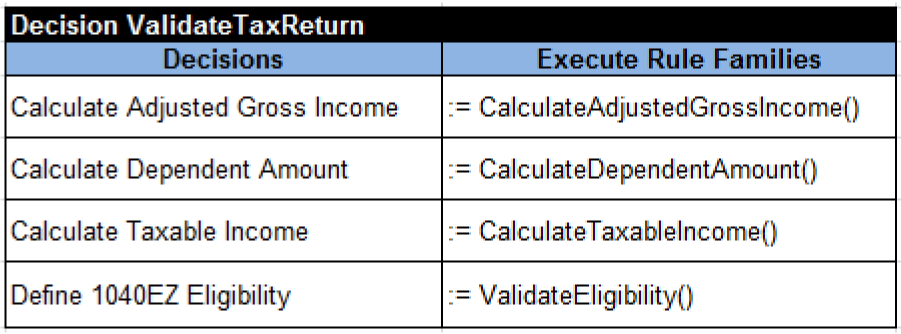
Let's keep going. As mentioned earlier, I noticed not one but two problems with your decision table. Are your decision variables "Taxable Income" and "Taxable Interest" already defined by the time when you try to validate them?

User. Oh no. They are parts of the calculation logic. Let me look again at the first page of 1040EZ… OK, Taxable Interest is defined in Line 2 and simply should be entered by a taxpayer. However, the Taxable Income is defined in Line 6 and it should be calculated. I am afraid we have to repeat the first 3 calculation steps before starting validation:

- Calculate Adjusted Gross Income
- Calculate Dependent Amount
- Calculate Taxable Income (Line 6)

OR. No problem, simply add these steps to your decision table ""ValidateTaxReturn".

User. Here it goes:



Decision ValidateTaxReturn

Decisions	Execute Rule Families
Calculate Adjusted Gross Income	:= CalculateAdjustedGrossIncome()
Calculate Dependent Amount	:= CalculateDependentAmount()
Calculate Taxable Income	:= CalculateTaxableIncome()
Define 1040EZ Eligibility	:= ValidateEligibility()

But why should we repeat these steps inside both decisions? Maybe it is better to add one more decision, say "Pre-Process", that should be executed before "ValidateTaxReturn".

OR. You certainly can do that, and in some cases it will be the right approach. However, I'd rather treat our two decisions "Validate" and "Calculate" as loosely coupled services and would rather minimize unnecessary dependencies. Besides, we will reuse the same calculation decision tables anyway. So, let's stick to what you've already done.

Using Formulas inside Decision Tables

User. OK. Now we have to put the actual calculation logic inside decision tables that we already have specified within our table "ValidateTaxReturn". The first decision table is "CalculateAdjustedGrossIncome". We will need to calculate the decision variable "Adjusted Gross Income" that is specified in Line 4 as a sum of Line 1 (Wages), Line 2 (Taxable Interest), and Line 3 (Unemployment Compensation). I can use a decision table with only one conclusion for the decision variable "Adjusted Gross Income". But how can I define a sum of different type variables?

OR. You can get access to all type variables using predefined get-methods. For example, assuming that the mentioned decision variables have real values, you may access them as:

```
getReal("Wages")
getReal("Taxable Interest")
getReal("Unemployment Compensation")
```

OpenRules® allows you to put any formulas inside decision table cells starting with "::=". Also do not forget to always put your formula in parenthesis. Let me help you for the first time:

DecisionTable CalculateAdjustedGrossIncome	
Conclusion	
Adjusted Gross Income	
Is	::= (getReal("Wages") + getReal("Taxable Interest") + getReal("Unemployment Compensation"))

User. I got it. But even if we do not use these decision variables as decision table columns, we probably still have to put them in our glossary later on.

OR. Yes, that is a good point. Before we jump to more complex calculation rules, let's do very similar decision table "CalculateTotalPayment".

User. It defines "Total Payments" as the sum of "Tax Withheld" and "Earned Income Credit". Here it goes.

DecisionTable CalculateTotalPayments	
Conclusion	
Total Payments	
Is	::= (getReal("Tax Withheld") + getReal("Earned Income Credit"))

OR. I am glad you did not forget parenthesis. While it would not matter in this case it is a good practice to always put your formulas in parenthesis.

Now, let's leave the most complex rule "CalculateDependentAmout" for desert, and next work on the decision table to "CalculateTaxableIncome".

When Conditions and Conclusions Share Decision Variables

OR. Decision table "CalculateTaxableIncome" does not simply uses a formula, but has an interesting flavor, if the formula for Taxable Income produces a negative result we need to set Taxable Income to 0. How would you approach this problem?

User. We may use a formula not only in a rule conclusion like we did earlier but also in its condition. We need to consider separate cases when a Boolean expression

```
getReal("Adjusted Gross Income") > getReal("Dependent Amount")
```

is TRUE and when it is FALSE.

OR. OpenRules® “DecisionTableExecuteTemplate” includes one more optional condition, “ConditionAny”, that allows you to handle such cases. Here is how it could be done:

DecisionTable CalculateTaxableIncome			
ConditionAny		Conclusion	
Condition		Taxable Income	
Is True	:= (getReal("Adjusted Gross Income") > getReal("Dependent Amount"))	Is	::= (getReal("Adjusted Gross Income") - getReal("Dependent Amount"))
Is False	:= (getReal("Adjusted Gross Income") > getReal("Dependent Amount"))	Is	0

User. I do not like this implementation – it seems that it calculates the same expression 3 times.

OR. You can create a special method that does it only once. Do you like the following implementation more?

```
Method double taxableIncome()
return getReal("Adjusted Gross Income") - getReal("Dependent Amount");
```

DecisionTable CalculateTaxableIncome			
ConditionAny		Conclusion	
Condition		Taxable Income	
Is True	:= (taxableIncome() > 0)	Is	::= taxableIncome()
Is False	:= (taxableIncome() > 0)	Is	0

User. Of course, I like this implementation more than the first one.

OR. Actually you can go further and replace the entire decision table with the following method:

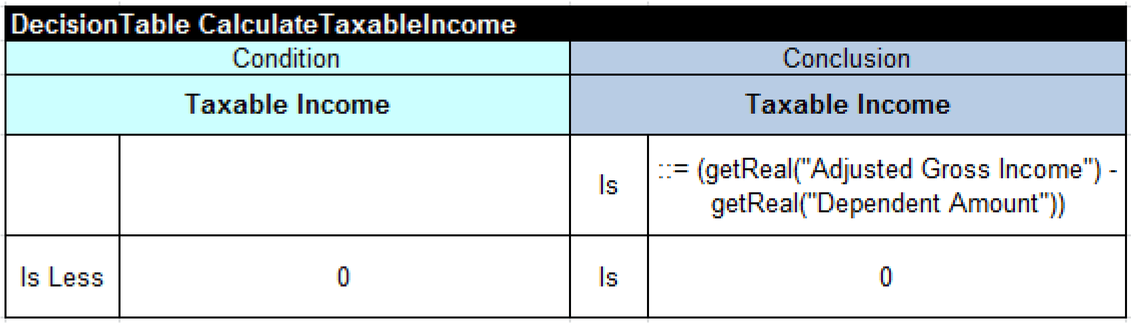
```
Method void CalculateTaxableIncome()
double ti = getReal("Adjusted Gross Income") - getReal("Dependent Amount");
if (ti > 0)
   setReal("Taxable Income", ti);
else
   setReal("Taxable Income", 0);
Log.info("Taxable Income: " + getReal("Taxable Income"));
```

In reality, OpenRules® treats decision tables as executable methods. However, while the above Java-like implementation will work, it is not recommended for two reasons:

1) it is difficult to understand (not mentioning to modify) for business users who usually are not familiar with Java

2) using Java methods you are losing the ability to automatically validate the consistency of your decision.

User. I think I have a better idea. What if we use "Taxable Income" as a decision variable for our condition? As you said earlier, we may rely on the fact that OpenRules® executes rules within a decision table in the top-down order. So, I can simplify our very first decision table as:



DecisionTable CalculateTaxableIncome			
Condition		Conclusion	
Taxable Income		Taxable Income	
		Is	::= (getReal("Adjusted Gross Income") - getReal("Dependent Amount"))
Is Less	0	Is	0

OR. It would be nice with one exception, the second condition that is supposed to nullify Taxable Income when it is less than 0 would never run. The problem is that a rule table of the type "DecisionTable" is implemented as an OpenRules® single-hit table. That means as soon as one rule is satisfied all other rules below it will be ignored.

User. But OpenRules® also supports multi-hit tables with rule overrides.

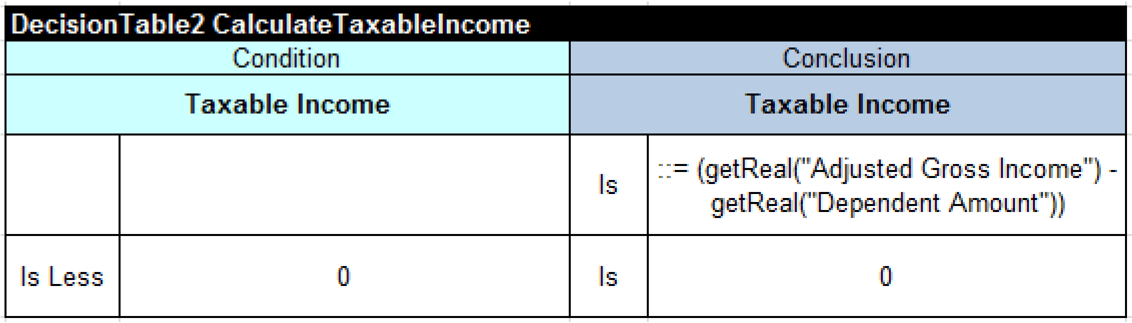
OR. That's true. Multi-hit tables execute all rules for which the conditions are satisfied.

User. Can we tell OpenRules® that we want to treat our particular decision table as a multi-hit table?

OR. Yes you can. Instead of the keyword "DecisionTable" you would use the keyword "DecisionTable1". But in your case it also would not help. The problem is that by default multi-hit tables first evaluate conditions for all rules and only then start executing actions for all satisfied rules. That means that changes in your decision variables produced by any actions within the same multi-hit rule table would not be taken into consideration when OpenRules® decides to execute this rule or not.

User. I'd prefer all possible options – my need seems to be reasonable and probably useful in many other cases.

OR. Do not be too upset – OpenRules® does provide these options. If you write "**DecisionTable2**" instead of the keyword "DecisionTable", OpenRules® will do exactly what you are looking for. It will execute all rules in the top-down order, evaluating rule conditions only after executing (or skipping) previous rules. So, here is another correctly working version of your “special” decision table:



DecisionTable2 CalculateTaxableIncome			
Condition		Conclusion	
Taxable Income		Taxable Income	
		Is	::= (getReal("Adjusted Gross Income") - getReal("Dependent Amount"))
Is Less	0	Is	0

User. I prefer this version.

OR. No problems. Now you understand the use of different types of decision tables: “**DecisionTable**”, “**DecisionTable1**”, and “**DecisionTable2**”.

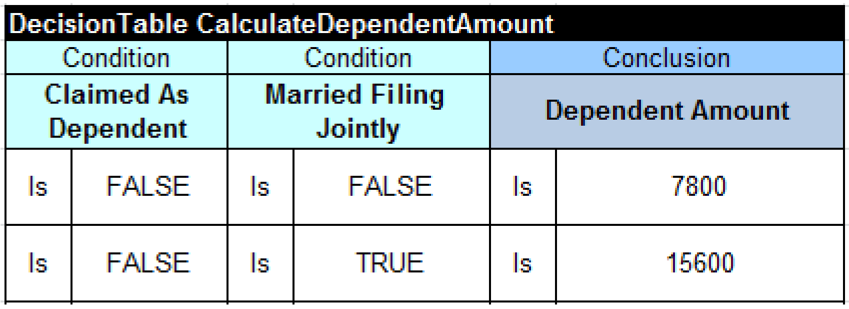
Let’s switch to more complex dependent rules from the back of the form 1040EZ. What does the form tells us about the Dependent Amount calculation?

More Complex Decision Tables

User. Line 5 asks a question, “Can your parents (or somebody else) claim you on their return?” If the answer is “No” then the Dependent Amount will be $7,800 if a taxpayer is single or $15,600 if married filing jointly. However, if the answer is “Yes” we should go to the second page…

OR. OK, to start with something, I suggest we create a decision table “CalculateDependentAmout” with the first condition “Claimed As Dependent” is FALSE. How would you address the “single” vs. “married filing jointly” situation?

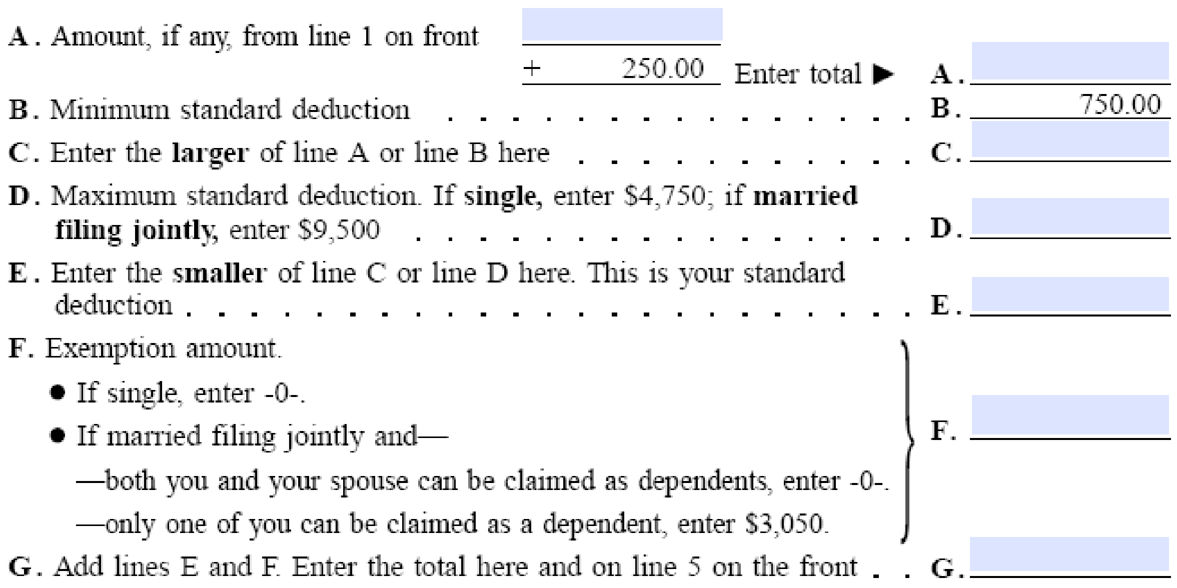
User. I don’t think we need a separate condition for a marital status – we may simple check condition for the decision variable “Married Filing Jointly”. Here is the proper decision table:



DecisionTable CalculateDependentAmount					
Condition		Condition		Conclusion	
Claimed As Dependent		Married Filing Jointly		Dependent Amount	
Is	FALSE	Is	FALSE	Is	7800
Is	FALSE	Is	TRUE	Is	15600

OR. Very good. Now we can consider all other situations described on the second page in the case when the first condition is TRUE.

User. Here is what the second page says:



A. Amount, if any, from line 1 on front

+ 250.00 Enter total ▶ A.

B. Minimum standard deduction . . . . B. 750.00

C. Enter the **larger** of line A or line B here . . . . C.

D. Maximum standard deduction. If **single**, enter \$4,750; if **married filing jointly**, enter \$9,500 . . . . D.

E. Enter the **smaller** of line C or line D here. This is your standard deduction . . . . E.

F. Exemption amount.

- If single, enter -0-.
- If married filing jointly and—

—both you and your spouse can be claimed as dependents, enter -0-.

—only one of you can be claimed as a dependent, enter \$3,050. F.

G. Add lines E and F. Enter the total here and on line 5 on the front . . G.

So, we need to calculate all Lines from A to F and then define the decision variable “Dependent Amount” (also Line G) as a sum of Lines E and F. I can add conclusions for Lines A, B, C, D, E, and F and then we consider additional conditions when we need them.

OR. Sounds like a good plan. Go for it.

User. OK, let me go line by line.

Line A can be calculated using the formula `::= (getReal(“Wages”) + 250)`

Line B is always a constant: 750.
Line C requires a maximum of Line A and Line B. Should I add conditions like A>B?

OR. Actually you do not have to. In your formula you can simply write:

```
::= Math.max( getReal(“Line A”), getReal(“Line B”))
```

Here we are using the standard Java method Math.max(x,y) that provide a maximum between x and y.

User. Nice! I like that OpenRules® allows us to use any standard Java method inside rules without any additional configuration.

Let's look at Line D. We already have a condition for the decision variable "Married Filing Jointly". So, if it is FALSE we will use the constant 4,750, and if it is TRUE we will use 9,500.

Line E is very similar to Line C but it deals with a minimum of two numbers. Probably I can use a similar formula:

```
::= Math.min( getReal("Line C"), getReal("Line D"))
```

OR. Perfect. Now let's evaluate Line F.

User. Here we have two sub-conditions when the decision variable "Married Filing Jointly" is TRUE:

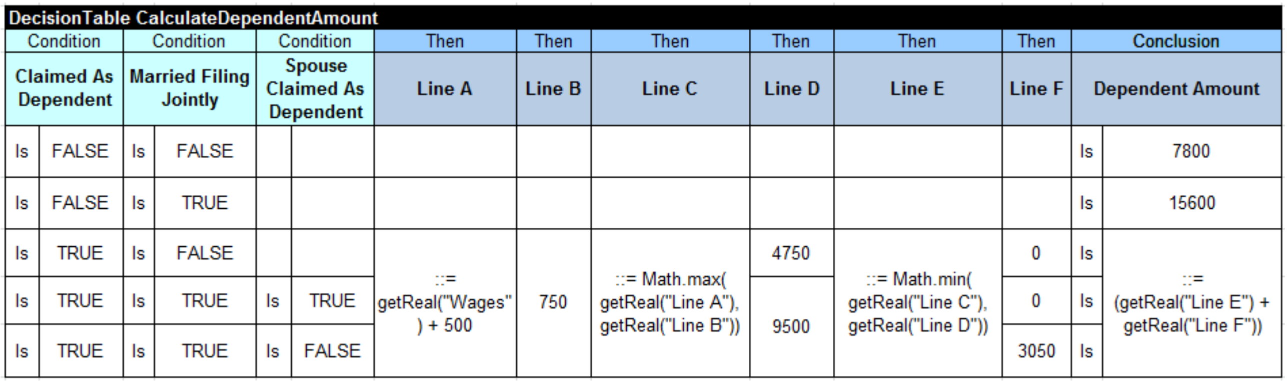
- both you and your spouse can be claimed as dependents, or
- only one of you can be claimed as a dependent.

We already have the decision variable for (I was) "Claimed As Dependent". Now we can define a new decision variable (my) "Spouse Claimed As Dependent" and consider different combinations of these two decision variables. Right?

Wait a minute! If I am already working with the worksheet for dependents it means the decision variable "Claimed As Dependent" is already TRUE. It also means that the first sub-condition should simple check if "Spouse Claimed As Dependent" is TRUE, and the second one – if it is FALSE. Why does Form 1040EZ ask a question in such a complicated form when they already know the answer?

OR. You are certainly pointing to a problem in probably one of the most used IRS forms, and in spite all attempts by the IRS to keep it as simple as possible they still made a mistake.

User. Maybe they should try to do what we are doing. Anyway, I think I am ready to extend my previous table. It's becoming very wide, it looks better in the Excel file, but here it is in a slightly compressed form:

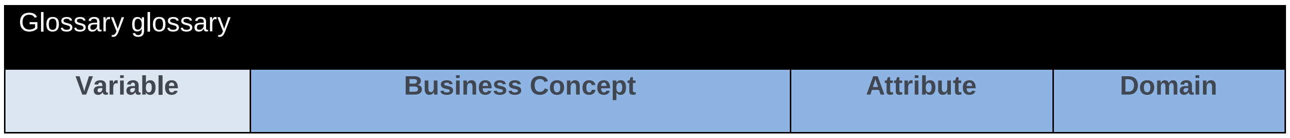


DecisionTable CalculateDependentAmount														
Condition		Condition		Condition		Then	Then	Then	Then	Then	Then	Conclusion	
Claimed As Dependent		Married Filing Jointly		Spouse Claimed As Dependent		Line A	Line B	Line C	Line D	Line E	Line F	Dependent Amount	
Is	FALSE	Is	FALSE									Is	7800
Is	FALSE	Is	TRUE									Is	15600
Is	TRUE	Is	FALSE			::= getReal("Wages") + 500	750	::= Math.max(getReal("Line A"), getReal("Line B"))	4750	::= Math.min(getReal("Line C"), getReal("Line D"))	0	Is	::= (getReal("Line E") + getReal("Line F"))
Is	TRUE	Is	TRUE	Is	TRUE				9500		0	Is	
Is	TRUE	Is	TRUE	Is	FALSE						3050	Is	

OR. Let's check how it will work. The rule 3 covers a situation when a taxpayer is claimed as a dependent but she/he is single (or at least did not file jointly with his/her spouse). Yes, your Line D will be 4750 and Line F will be 0. That's correct. The next rules states that when a taxpayer is claimed as a dependent, married filing jointly is TRUE, and the spouse is also claimed as a dependent, then Line D will be 9500 and Line F will be 0. And finally the last rule is similar to a previous rule but covers the case when only the main taxpayer can be claimed as a dependent. In this case while Line D should stay the same, Line F should show 3050. It seems that this is a good representation of a complex set of conditions. I also like the way how you merged cells for the same values to avoid repeating the same formulas and values.

Defining a Business Glossary

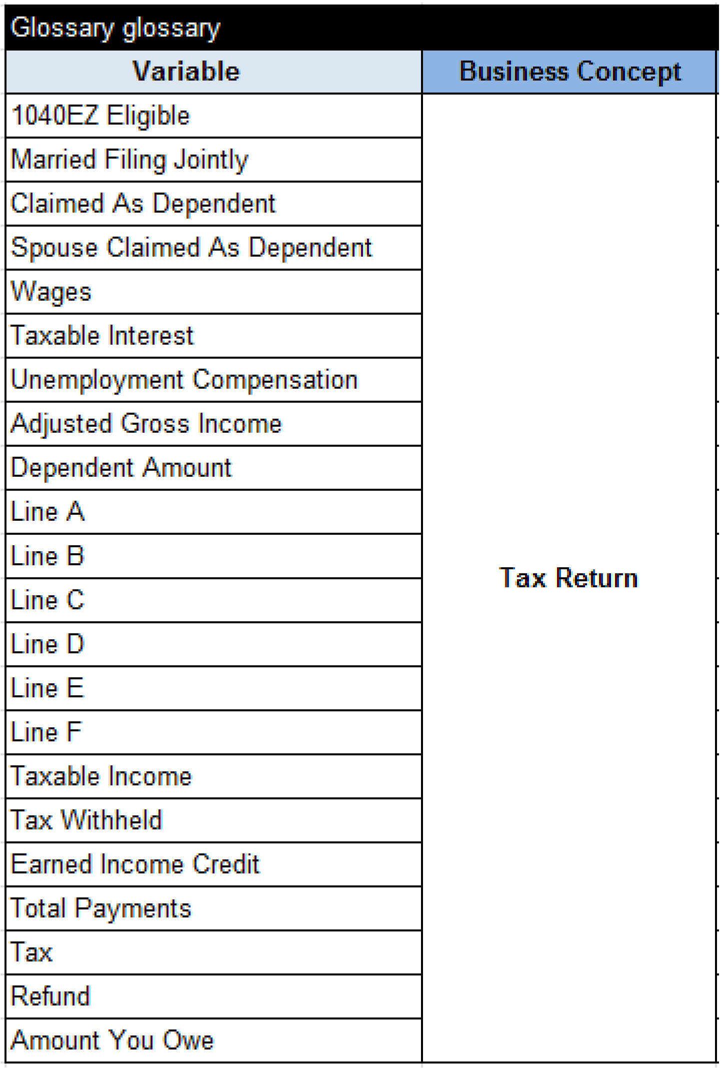
OR. So, let's briefly summarize what we've already accomplished. We started with decisions and they led us to specifications of different decision tables. Inside the decision tables we freely introduced different decision variables assuming that they are somehow defined. Now, when we have all of the decision tables in place, it is time to actually define our decision variables. To do this we simply need to put all the decision variables in a Glossary table that has the following structure:



Glossary glossary			
Variable	Business Concept	Attribute	Domain

The first column will simply list all decision variables using exactly the same name that we used inside the decision tables. Then we have to associate different decision variables with different business concepts. It seems in our case we need use only one business concept "Tax Return" because it includes all of the input decision variables, like Wages or Married Filing Jointly, all intermediate decision variables such as Dependent Amount, and two output decision variables Return and Amount You Owe.

User. OK, here is the glossary table with the first two columns filled out:



Glossary glossary	
Variable	**Business Concept**
1040EZ Eligible	**Tax Return**
Married Filing Jointly	
Claimed As Dependent	
Spouse Claimed As Dependent	
Wages	
Taxable Interest	
Unemployment Compensation	
Adjusted Gross Income	
Dependent Amount	
Line A	
Line B	
Line C	
Line D	
Line E	
Line F	
Taxable Income	
Tax Withheld	
Earned Income Credit	
Total Payments	
Tax	
Refund	
Amount You Owe	

OR. Good, I hope you did not forget any previously mentioned decision variable. But even if you did OpenRules® will point us to such errors. It is good that you merged all rows in the second column showing that all decision variables belong to the concept Tax Return. Now we need to fill out the remaining two columns. The column "Attribute" contains "technical" names of decision variables – these names will be used to connect our decision variables with attributes of objects used by actual applications that will use our decision "Apply1040EZ". The application objects could be defined in Java, in Excel tables (like we will do in the next section), in XML, etc. The decision does not have to know about it, the only requirement is that the attribute names should follow the usual naming convention for identification in languages like Java, it basically means no spaces allowed. The last column "Domain" is optional but it can be useful to specify which values are allowed to be used for different decision variables. In our case the first 4 decision variables are Boolean, so we may define their domain as "TRUE, FALSE", all other decision variables are real numbers. Here is our final glossary:

Glossary glossary			
Variable	**Business Concept**	**Attribute**	**Domain**
1040EZ Eligible	**Tax Return**	eligible	TRUE,FALSE
Married Filing Jointly		marriedFilingJointly	TRUE,FALSE
Claimed As Dependent		claimedAsDependent	TRUE,FALSE
Spouse Claimed As Dependent		spouseClaimedAsDependent	TRUE,FALSE
Wages		wages	
Taxable Interest		taxableInterest	
Unemployment Compensation		unemploymentCompensation	
Adjusted Gross Income		adjustedGrossIncome	
Dependent Amount		dependentAmount	
Line A		dependentLineA	
Line B		dependentLineB	
Line C		dependentLineC	
Line D		dependentLineD	
Line E		dependentLineE	
Line F		dependentLineF	
Taxable Income		taxableIncome	
Tax Withheld		taxWithheld	
Earned Income Credit		earnedIncomeCredit	
Total Payments		totalPayments	
Tax		tax	
Refund		refund	
Amount You Owe		amountYouOwe	

Do not forget that all columns in the very first row should be merged.

User. OK, but how will we associate our Tax Return with actual tax returns?

OR. We may assume that the actual Tax Return will be put in our decision under the name "taxReturn". Here is a table DecisionObject that associates our business concept Tax Return with a Java object that we may get from the decision:

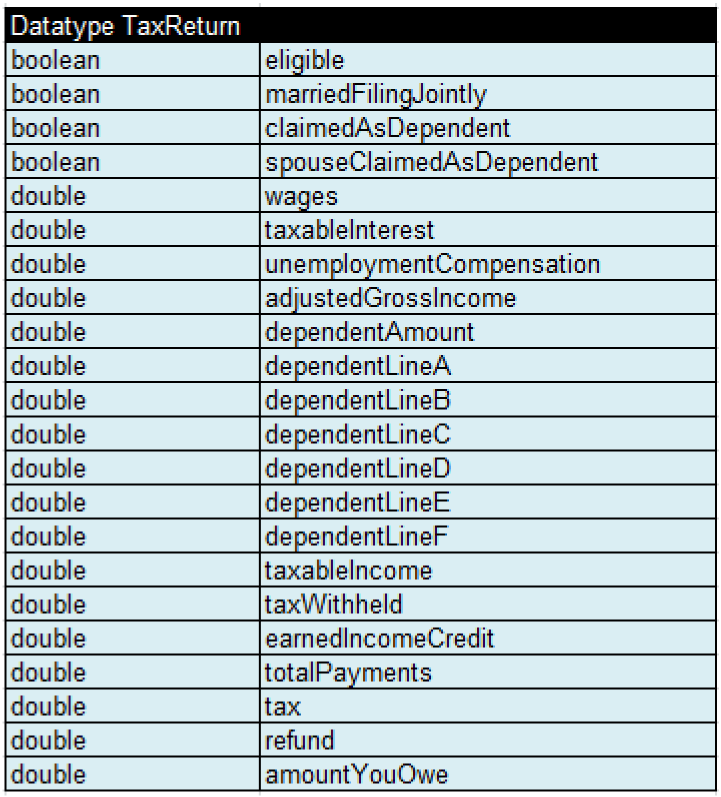
DecisionObject decisionObjects	
Business Concept	**Business Object**
Tax Return	:= decision.get("taxReturn")

User. But can we test our decision before we integrate it with an actual Java application? I do not know (and I do not want to know) how to create a Java object "taxReturn".

OR. Yes, we can define our own test data directly in Excel using OpenRules® Datatype and Data tables.

Defining Test Data in Excel

OR. First, we define a Datatype "TaxReturn":



Datatype TaxReturn	
boolean	eligible
boolean	marriedFilingJointly
boolean	claimedAsDependent
boolean	spouseClaimedAsDependent
double	wages
double	taxableInterest
double	unemploymentCompensation
double	adjustedGrossIncome
double	dependentAmount
double	dependentLineA
double	dependentLineB
double	dependentLineC
double	dependentLineD
double	dependentLineE
double	dependentLineF
double	taxableIncome
double	taxWithheld
double	earnedIncomeCredit
double	totalPayments
double	tax
double	refund
double	amountYouOwe

Please note that we use exactly the same attribute names as in out glossary. You can use the following types for your attributes:

- String – for text attributes
- int – for integer numbers
- Boolean – for Boolean attributes
- double – for real numbers
- Date – for dates

By the way they correspond to the standard Java types and are case-sensitive.

Now we will create several test instances of the type TaxReturn:

Data TaxReturn taxReturns			
eligible	**Eligible**	FALSE	FALSE
marriedFilingJointly	**MarriedFilingJointly**	TRUE	TRUE
claimedAsDependent	**claimedAsDependent**	TRUE	TRUE
spouseClaimedAsDependent	**spouseClaimedAsDependent**	TRUE	TRUE
wages	**wages**	32026	82026
taxableInterest	**taxableInterest**	1450	1450
unemploymentCompensation	**unemploymentCompensation**	0	0
adjustedGrossIncome	**adjustedGrossIncome**	0	0
dependentAmount	**dependentAmount**	0	0
dependentLineA	**dependentLineA**	0	0
dependentLineB	**dependentLineB**	0	0
dependentLineC	**dependentLineC**	0	0
dependentLineD	**dependentLineD**	0	0
dependentLineE	**dependentLineE**	0	0
dependentLineF	**dependentLineF**	0	0
taxableIncome	**taxableIncome**	0	0
taxWithheld	**taxWithheld**	4530	4530
earnedIncomeCredit	**earnedIncomeCredit**	300	300
totalPayments	**totalPayments**	0	0
tax	**tax**	0	0
refund	**refund**	0	0
amountYouOwe	**amountYouOwe**	0	0

To make a test instance available for any Java code that is going to be used for testing our decision, we will add a simple method "getTaxReturn" that returns the first test instance (with index 0).

Method TaxReturn getTaxReturn()
return taxReturns[0];

User. I see how you created a new Datatype TaxReturn and two test cases. I can make more test cases by simply adding more columns to the table "taxReturns". But I am not sure why we are going to use this method "getTaxReturn".

OR. In many cases you do not need such methods and can directly connect your Excel-based test cases with your decision (see how it was done in examples described in the document "Getting Started"). However, here I also wanted to demonstrate to you how to organize a more sophisticated interface between an Excel-based decision and the Java program that will execute it.

Decision Execution

Now we are ready to execute the defined decision against the above test data. The project "Decision1040EZ" includes the following Java launcher for our decision.

```
public class Main {

      public static void main(String[] args) {
            String fileName = "file:rules/main/Decision.xls";
            OpenRulesEngine engine = new OpenRulesEngine(fileName);
            Decision decision = new Decision("Apply1040EZ",engine);
            DynamicObject taxReturn =
                 (DynamicObject) engine.run("getTaxReturn");
            engine.log("=== INPUT:\n" + taxReturn);
            decision.put("taxReturn",taxReturn);
            decision.execute();
            engine.log("=== OUTPUT:\n" + taxReturn);
      }
}
```

This code first creates an instance "engine" of the standard OpenRules class "OpenRulesEngine" based on our main Excel file "Decision.xls" (placed in the folder "rules/main"). It also creates a decision "Apply1040EZ" based on this "engine".

Then the engine reads our test data using the above method "getTaxReturn", saves it as a DynamicObject "taxReturn", and prints it out as an INPUT. When we run the code it will print out test data as follows:

```
=== INPUT:
TaxReturn(id=0) {
  adjustedGrossIncome=0.0
  amountYouOwe=0.0
  claimedAsDependent=true
  dependentAmount=0.0
  dependentLineA=0.0
  dependentLineB=0.0
  dependentLineC=0.0
  dependentLineD=0.0
  dependentLineE=0.0
  dependentLineF=0.0
  earnedIncomeCredit=300
  eligible=false
  marriedFilingJointly=true
  refund=0.0
  spouseClaimedAsDependent=true
  tax=0.0
  taxWithheld=4,530
  taxableIncome=0.0
  taxableInterest=1,450
  totalPayments=0.0
```

```
    unemploymentCompensation=0.0
    wages=32,026
  }
```

Then it adds the object taxReturn" to the decision,

```
decision.put("taxReturn",taxReturn);
```

and executes the decision:

```
decision.execute();
```

During the execution the following trace will be produced.

```
*** Decision Apply1040EZ ***
Decision has been initialized
Decision Apply1040EZ: Validate
   Decision ValidateTaxReturn: Calculate Adjusted Gross Income
   Conclusion: Adjusted Gross Income Is 33476.0
   Decision ValidateTaxReturn: Calculate Dependent Amount
   Line A := 32026.0500
   Line B := 750
   Line C := 32026.05
   Line D := 9500
   Line E := 9500.0
   Line F := 0
   Conclusion: Dependent Amount Is 9500.0
   Decision ValidateTaxReturn: Calculate Taxable Income
   Conclusion: Taxable Income Is 23976.0
   Decision ValidateTaxReturn: Define 1040EZ Eligibility
   Conclusion: 1040EZ Eligible Is true
   YOU CAN USE 1040EZ FORM from ValidateEligibility
Decision Apply1040EZ: Calculate
   Decision DetermineTaxReturn: Calculate Adjusted Gross Income
   Conclusion: Adjusted Gross Income Is 33476.0
   Decision DetermineTaxReturn: Calculate Dependent Amount
   Line A := 32026.0500
   Line B := 750
   Line C := 32026.05
   Line D := 9500
   Line E := 9500.0
   Line F := 0
   Conclusion: Dependent Amount Is 9500.0
   Decision DetermineTaxReturn: Calculate Taxable Income
   Conclusion: Taxable Income Is 23976.0
   Decision DetermineTaxReturn: Calculate Total Payments
   Conclusion: Total Payments Is 4830.0
   Decision DetermineTaxReturn: Define Tax using the standard Tax Table
   Decision DetermineTaxReturn: Calculate Refund
   Conclusion: Refund Is 1934.0
   Conclusion: Amount You Owe Is 0
```

And finally, this code will print a modified test object "taxReturn" as an OUTPUT.

```
=== OUTPUT:
TaxReturn(id=0) {
  adjustedGrossIncome=33,476
  amountYouOwe=0.0
  claimedAsDependent=true
  dependentAmount=9,500
  dependentLineA=32,026.05
  dependentLineB=750
  dependentLineC=32,026.05
  dependentLineD=9,500
  dependentLineE=9,500
  dependentLineF=0.0
  earnedIncomeCredit=300
  eligible=true
  marriedFilingJointly=true
  refund=1,934
  spouseClaimedAsDependent=true
  tax=2,896
  taxWithheld=4,530
  taxableIncome=23,976
  taxableInterest=1,450
  totalPayments=4,830
  unemploymentCompensation=0.0
  wages=32,026
}
```

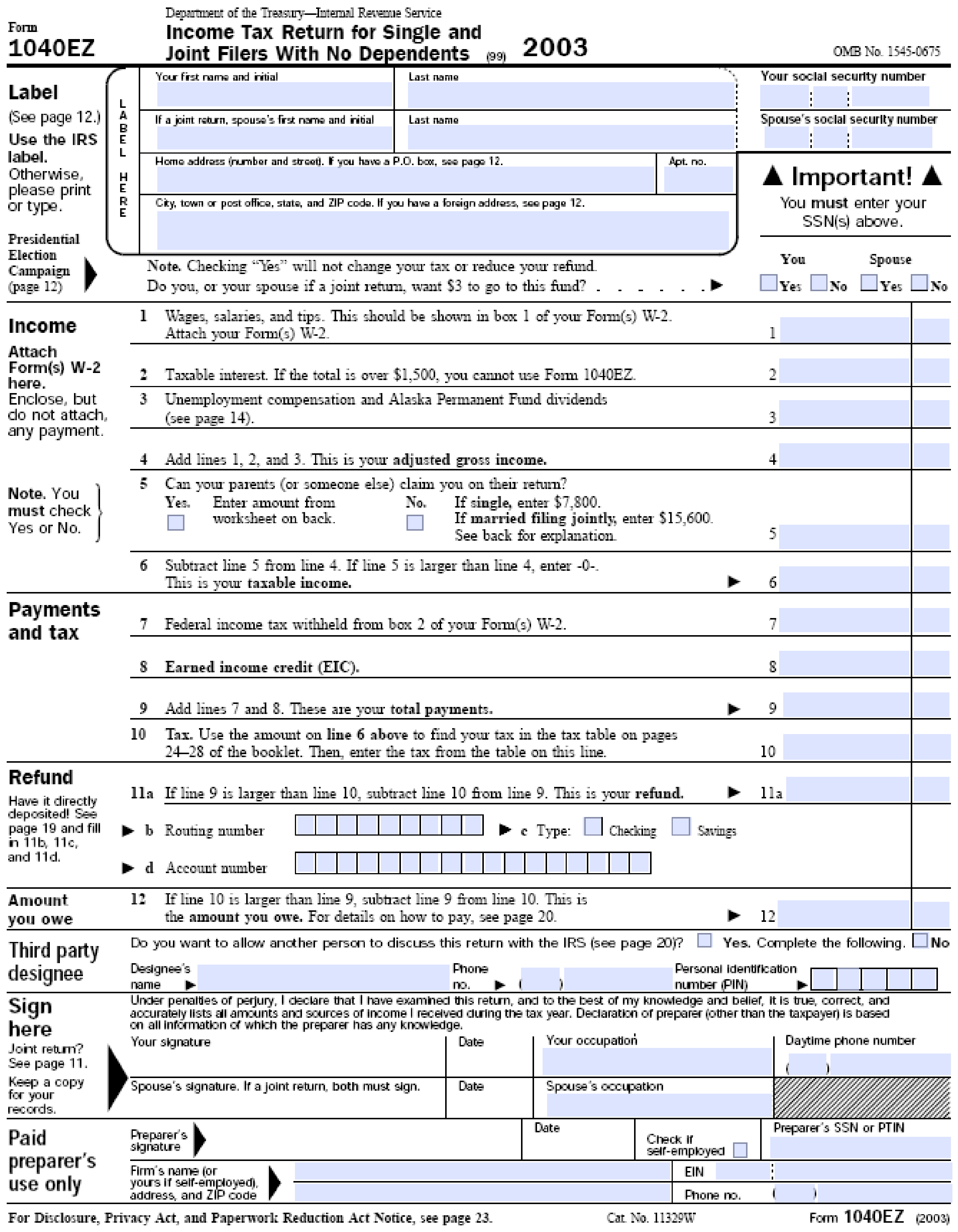
When you decide to pass your already well-tested decision to your IT people, they will simply replace the object "taxReturn" with real Java objects. They may use their own class TaxReturn. The only integration requirement is that the names of attributes in their classes should be the same as in our Glossary. An important point is that your IT guys do not have to even look at your decision and decision tables.

This completes the decision part of our tutorial.

User. This decision was not as simple as the examples from "Getting Started" but it gave me many more choices that I can use in real-world decision. Thank you.

OR. You are welcome. You can find many more details in the User Manual. You also may consider a completely opposite approach to the 1040EZ implementation using OpenRules® Dialog or ORD™ – see http://openrules.com/pdf/Tutorial.Dialog1040EZ.pdf.

Appendix. Form 1040EZ



Form **1040EZ**

Department of the Treasury—Internal Revenue Service

Income Tax Return for Single and Joint Filers With No Dependents (99) **2003**

OMB No. 1545-0675

Label (See page 12.) **Use the IRS label.** Otherwise, please print or type.

LABEL HERE

Your first name and initial | Last name | **Your social security number**

If a joint return, spouse's first name and initial | Last name | **Spouse's social security number**

Home address (number and street). If you have a P.O. box, see page 12. | Apt. no.

City, town or post office, state, and ZIP code. If you have a foreign address, see page 12.

▲ Important! ▲ You **must** enter your SSN(s) above.

Presidential Election Campaign (page 12) **Note.** Checking "Yes" will not change your tax or reduce your refund. Do you, or your spouse if a joint return, want $3 to go to this fund? ▶ **You** ☐ Yes ☐ No **Spouse** ☐ Yes ☐ No

Income

Attach Form(s) W-2 here. Enclose, but do not attach, any payment.

1	Wages, salaries, and tips. This should be shown in box 1 of your Form(s) W-2. Attach your Form(s) W-2.	1	
2	Taxable interest. If the total is over $1,500, you cannot use Form 1040EZ.	2	
3	Unemployment compensation and Alaska Permanent Fund dividends (see page 14).	3	
4	Add lines 1, 2, and 3. This is your **adjusted gross income.**	4	
5	Can your parents (or someone else) claim you on their return? **Yes.** ☐ Enter amount from worksheet on back. **No.** ☐ If **single,** enter $7,800. If **married filing jointly,** enter $15,600. See back for explanation.	5	
6	Subtract line 5 from line 4. If line 5 is larger than line 4, enter -0-. This is your **taxable income.** ▶	6	

Note. You **must** check Yes or No.

Payments and tax

7	Federal income tax withheld from box 2 of your Form(s) W-2.	7	
8	**Earned income credit (EIC).**	8	
9	Add lines 7 and 8. These are your **total payments.** ▶	9	
10	**Tax.** Use the amount on **line 6 above** to find your tax in the tax table on pages 24–28 of the booklet. Then, enter the tax from the table on this line.	10	

Refund

Have it directly deposited! See page 19 and fill in 11b, 11c, and 11d.

11a	If line 9 is larger than line 10, subtract line 10 from line 9. This is your **refund.** ▶	11a	
▶ b	Routing number ☐☐☐☐☐☐☐☐☐ ▶ c Type: ☐ Checking ☐ Savings		
▶ d	Account number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐		

Amount you owe

12	If line 10 is larger than line 9, subtract line 9 from line 10. This is the **amount you owe.** For details on how to pay, see page 20. ▶	12	

Third party designee Do you want to allow another person to discuss this return with the IRS (see page 20)? ☐ **Yes.** Complete the following. ☐ **No**

Designee's name ▶ Phone no. ▶ ( ) Personal identification number (PIN) ▶ ☐☐☐☐☐

Sign here Joint return? See page 11. Keep a copy for your records.

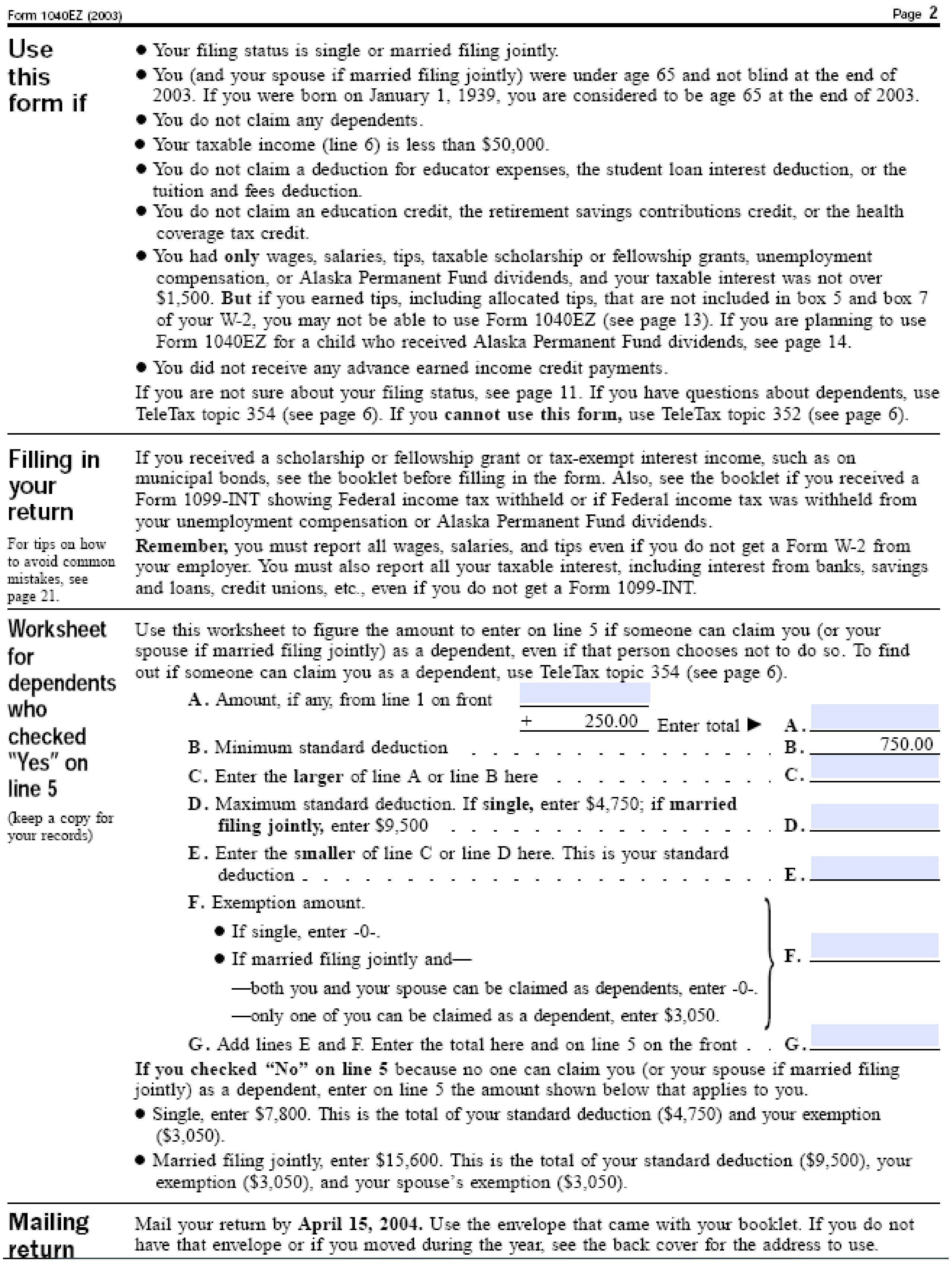
Under penalties of perjury, I declare that I have examined this return, and to the best of my knowledge and belief, it is true, correct, and accurately lists all amounts and sources of income I received during the tax year. Declaration of preparer (other than the taxpayer) is based on all information of which the preparer has any knowledge.

Your signature	Date	Your occupation	Daytime phone number ( )
Spouse's signature. If a joint return, both must sign.	Date	Spouse's occupation	▨

Paid preparer's use only

Preparer's signature ▶	Date	Check if self-employed ☐	Preparer's SSN or PTIN
Firm's name (or yours if self-employed), address, and ZIP code ▶		EIN	
		Phone no.	( )

For Disclosure, Privacy Act, and Paperwork Reduction Act Notice, see page 23. Cat. No. 11329W Form **1040EZ** (2003)



Form 1040EZ (2003) Page 2

Use this form if

- Your filing status is single or married filing jointly.
- You (and your spouse if married filing jointly) were under age 65 and not blind at the end of 2003. If you were born on January 1, 1939, you are considered to be age 65 at the end of 2003.
- You do not claim any dependents.
- Your taxable income (line 6) is less than $50,000.
- You do not claim a deduction for educator expenses, the student loan interest deduction, or the tuition and fees deduction.
- You do not claim an education credit, the retirement savings contributions credit, or the health coverage tax credit.
- You had **only** wages, salaries, tips, taxable scholarship or fellowship grants, unemployment compensation, or Alaska Permanent Fund dividends, and your taxable interest was not over $1,500. **But** if you earned tips, including allocated tips, that are not included in box 5 and box 7 of your W-2, you may not be able to use Form 1040EZ (see page 13). If you are planning to use Form 1040EZ for a child who received Alaska Permanent Fund dividends, see page 14.
- You did not receive any advance earned income credit payments.

If you are not sure about your filing status, see page 11. If you have questions about dependents, use TeleTax topic 354 (see page 6). If you **cannot use this form,** use TeleTax topic 352 (see page 6).

Filling in your return

For tips on how to avoid common mistakes, see page 21.

If you received a scholarship or fellowship grant or tax-exempt interest income, such as on municipal bonds, see the booklet before filling in the form. Also, see the booklet if you received a Form 1099-INT showing Federal income tax withheld or if Federal income tax was withheld from your unemployment compensation or Alaska Permanent Fund dividends.

Remember, you must report all wages, salaries, and tips even if you do not get a Form W-2 from your employer. You must also report all your taxable interest, including interest from banks, savings and loans, credit unions, etc., even if you do not get a Form 1099-INT.

Worksheet for dependents who checked "Yes" on line 5

(keep a copy for your records)

Use this worksheet to figure the amount to enter on line 5 if someone can claim you (or your spouse if married filing jointly) as a dependent, even if that person chooses not to do so. To find out if someone can claim you as a dependent, use TeleTax topic 354 (see page 6).

A. Amount, if any, from line 1 on front	+ 250.00 Enter total ▶	**A.**	
B. Minimum standard deduction		**B.**	750.00
C. Enter the **larger** of line A or line B here		**C.**	
D. Maximum standard deduction. If **single,** enter $4,750; **if married filing jointly,** enter $9,500		**D.**	
E. Enter the **smaller** of line C or line D here. This is your standard deduction		**E.**	
F. Exemption amount. • If single, enter -0-. • If married filing jointly and— —both you and your spouse can be claimed as dependents, enter -0-. —only one of you can be claimed as a dependent, enter $3,050.		**F.**	
G. Add lines E and F. Enter the total here and on line 5 on the front		**G.**	

If you checked "No" on line 5 because no one can claim you (or your spouse if married filing jointly) as a dependent, enter on line 5 the amount shown below that applies to you.

- Single, enter $7,800. This is the total of your standard deduction ($4,750) and your exemption ($3,050).
- Married filing jointly, enter $15,600. This is the total of your standard deduction ($9,500), your exemption ($3,050), and your spouse's exemption ($3,050).

Mailing return

Mail your return by **April 15, 2004.** Use the envelope that came with your booklet. If you do not have that envelope or if you moved during the year, see the back cover for the address to use.