**Практическая работа №4.**

**Разработка расчетной подсистемы**

**Цель работы:** освоить основные *механизмы* 1С:Предприятие 8.2., используемые при решении расчетных задач

**Постановка задачи**

Подсистемы, связанные с расчетом заработной платы, в конфигурациях, разработанных на 1С:Предприятие 8, проектируются с использованием специализированных объектов, в центре которых стоят *планы видов расчета* и *регистры расчета*. Эти задачи считаются одними из самых сложных и трудоемких. Не случайно, для типовых конфигураций, характерна ситуация, когда *конфигурация* для ведения бухгалтерского учета содержит лишь минимальный набор "зарплатных" возможностей, которые рассчитаны лишь на отражение в учете зарплаты, налогов, отчислений. Причем, в таких конфигурациях *регистры расчета* обычно не используются. Бухгалтерскую конфигурацию, традиционно, дополняет *конфигурация*, ориентированная исключительно на учет заработной платы, в ней все уже построено вокруг упомянутых объектов, и, кроме того, разработана большая "*инфраструктура*", которая позволяет учесть особенности работы с зарплатой.

Мы, в нашей учебной конфигурации, разработаем подсистему учета заработной платы, основанную на регистрах расчета и планах *видов расчета*, а так же создадим некоторые вспомогательные *механизмы*.

Работа нашей подсистемы, если описать ее схематично, будет выглядеть следующим образом.

Когда приходит время начислять заработную плату, используется документ **НачислениеЗарплаты**, который заполняется вручную. В реально используемых "зарплатных" конфигурациях подобный документ обычно заполняется автоматически, на основе списков начислений и удержаний, закрепленных за сотрудниками. Обычно после автоматического заполнения *пользователь* имеет возможность отредактировать набор начислений и удержаний, отраженных в документе, а так же – их параметров.

После заполнения документа данные из него переносятся в *регистры расчета* **ОсновныеНачисления** и **ДополнительныеНачисления**. В регистре **ОсновныеНачисления** отражаются те виды начислений, которые, помимо прочих характеристик, можно описать периодом действия – то есть такие, как различные виды окладов, прогул (нулевое начисление за рабочие дни), отпуск. Важная особенность таких начислений – *вытеснение* одних видов другими. Например, начисление прогула вытесняет начисление оклада. В регистре **ДополнительныеНачисления** отражаются начисления, основная характеристика которых – это сумма начисления (например – материальная помощь, надбавка, штраф и т.д.) или некоторый *процент* от других начислений (различные виды премий).

После переноса сведений в регистры расчета производится расчет начислений. Расчет производится разными способами для разных начислений, причем, начисления характеризуются не только способом, но и очередностью расчета. Например, если **Премия** зависит от **Оклада** и от **Надбавки**, а **Надбавка**, в свою *очередь*, зависит от **Оклада**, то для правильного расчета **Премии** необходимо сначала рассчитать **Оклад**, потом – **Надбавку**, и уже потом – **Премию**.

Нужно учитывать и тот факт, что возможна ситуация, когда начисления предыдущих периодов нужно откорректировать. Если говорить о реальных конфигурациях, то суммы, начисленные в предыдущих периодах, обычно не правят (с них уже начислены налоги, произведены отчисления в фонды и т.д.), вместо этого используются так называемые сторно-записи (для учета отрицательных отклонений, например, если сотруднику неправильно начислили заработную плату, необоснованно увеличив ее). В случае, если работнику начислили меньше, чем положено, начисление в текущем периоде может быть увеличено.

Кроме того, нужно предусмотреть ситуацию в текущем периоде, когда изменение одних *видов расчета* приводит к необходимости пересчитать некоторые записи – автоматизировать этот процесс можно с помощью объекта **Перерасчет**.

В реально используемых конфигурациях, ориентированных на расчет зарплаты, расчетные *механизмы* дополняются большим набором вспомогательных объектов. Так, помимо начисления заработной платы нужно рассчитывать налоги и взносы, нужно вести учет взаиморасчетов с работниками, фиксировать в системе выплату заработной платы различными способами (наличными, путем банковского перечисления) с формированием всей необходимой документации. Кроме того, в подобных конфигурациях ведется учет сотрудников, плановые начисления и удержания вводятся с помощью специальных документов.

Наша подсистема расчета заработной платы будет включать в себя следующие объекты:

Справочники

1. ГрафикиРаботы
2. Сотрудники

Перечисления

1. СпособыРасчета
2. ОчередностиРасчета

Документы:

1. НачислениеЗаработнойПлаты

*Регистры сведений*:

1. РабочиеДниПоГрафику

*Регистры расчета*;

1. ОсновныеНачисления
2. ДополнительныеНачисления

*Планы видов расчета*;

1. ОсновныеНачисления
2. ДополнительныеНачисления

Обработки

1. ЗаполнениеГрафиковРаботы

Отчеты

1. ОсновныеНачисления (диаграмма Ганта)
2. Перерасчеты

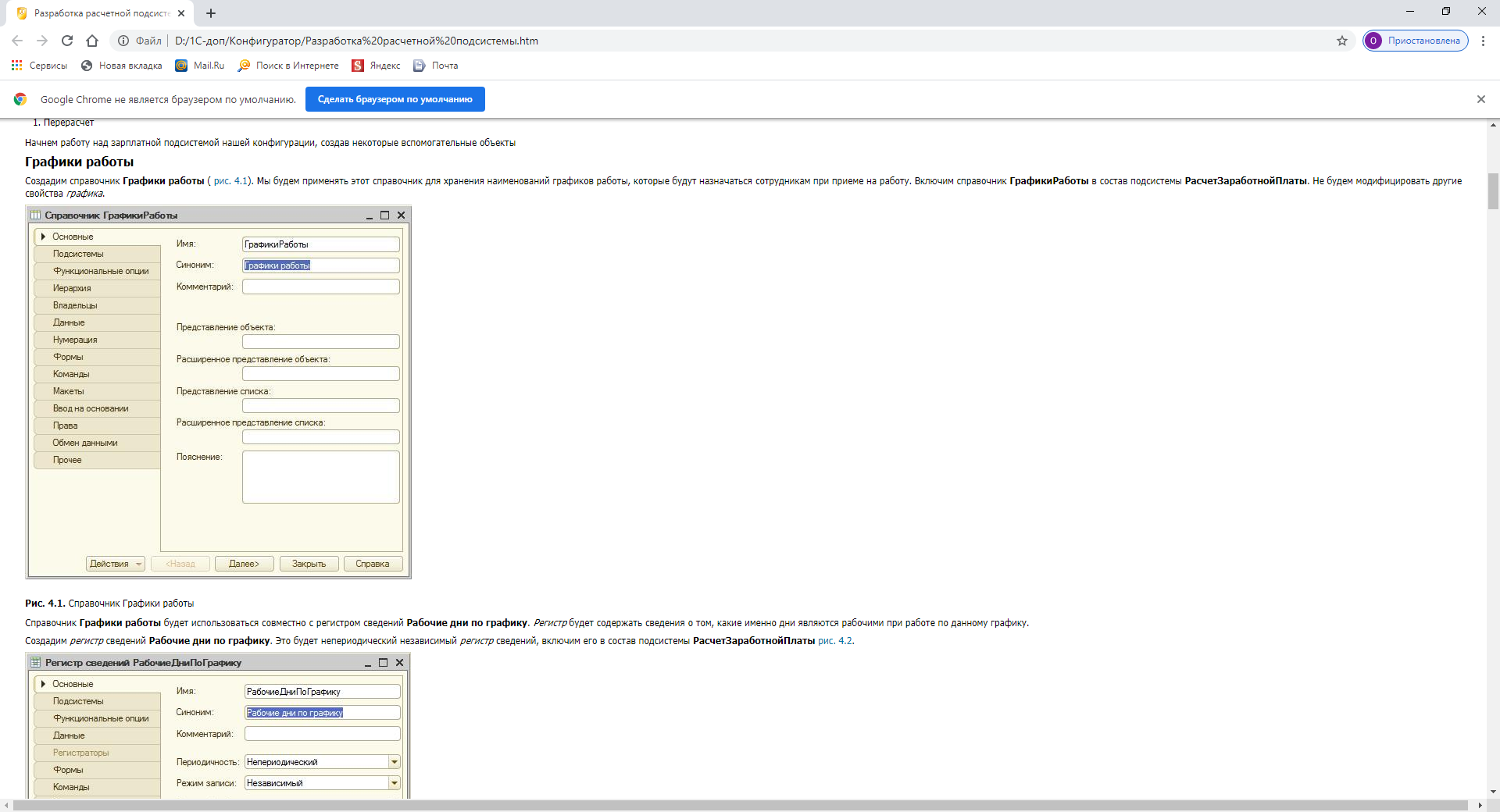
Перерасчеты

1. Перерасчет

Начнем работу над зарплатной подсистемой нашей конфигурации, создав некоторые вспомогательные объекты

**Графики работы**

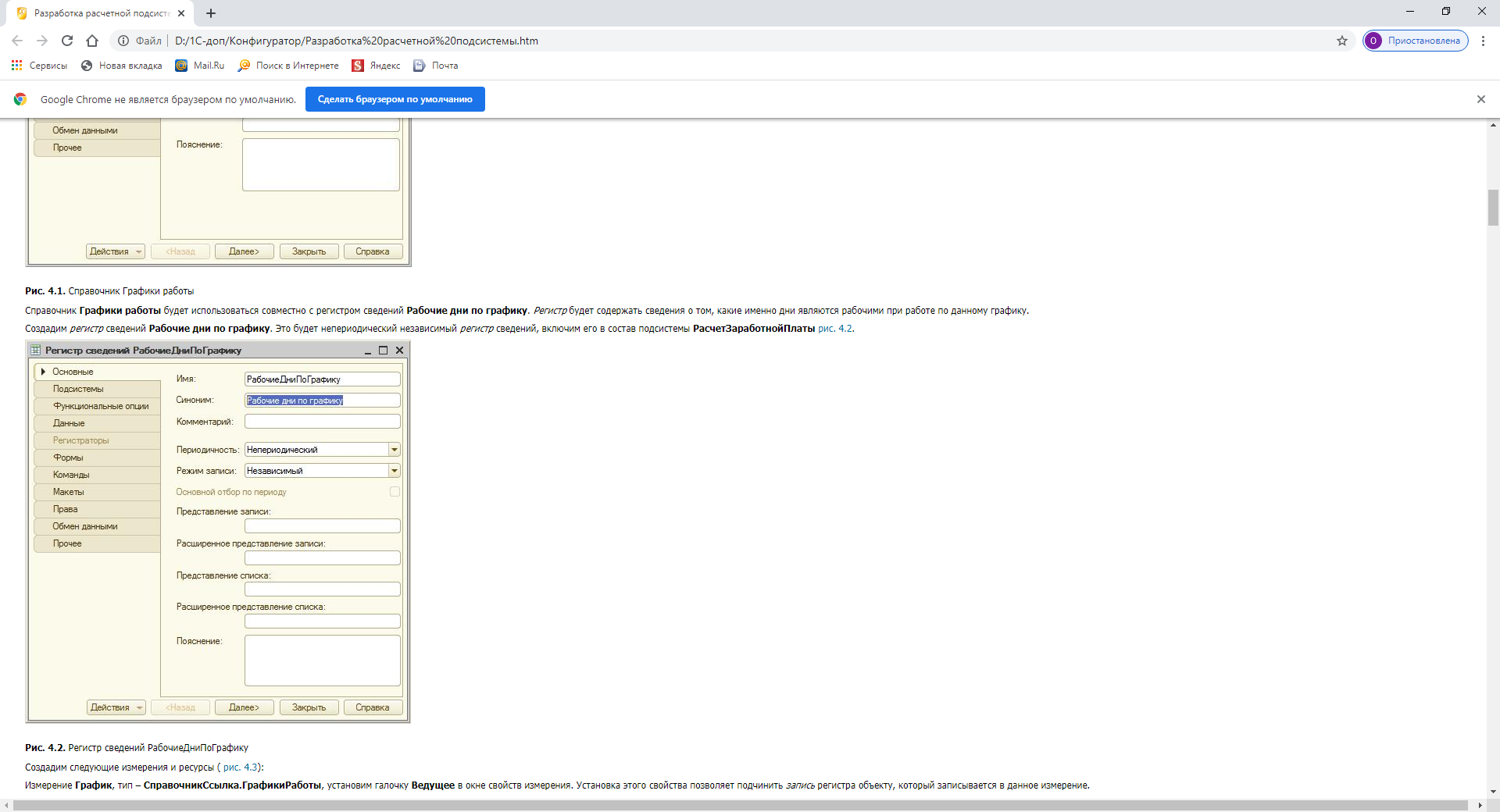
Создадим справочник **Графики работы** ([рис. 4.1](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=2#image.4.1)). Мы будем применять этот справочник для хранения наименований графиков работы, которые будут назначаться сотрудникам при приеме на работу. Включим справочник **ГрафикиРаботы** в состав подсистемы **РасчетЗаработнойПлаты**. Не будем модифицировать другие свойства *графика*.



**Рис. 4.1.**Справочник Графики работы

Справочник **Графики работы** будет использоваться совместно с регистром сведений **Рабочие дни по графику**. *Регистр* будет содержать сведения о том, какие именно дни являются рабочими при работе по данному графику.

Создадим *регистр* сведений **Рабочие дни по графику**. Это будет непериодический независимый *регистр* сведений, включим его в состав подсистемы **РасчетЗаработнойПлаты** [рис. 4.2](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=2#image.4.2).



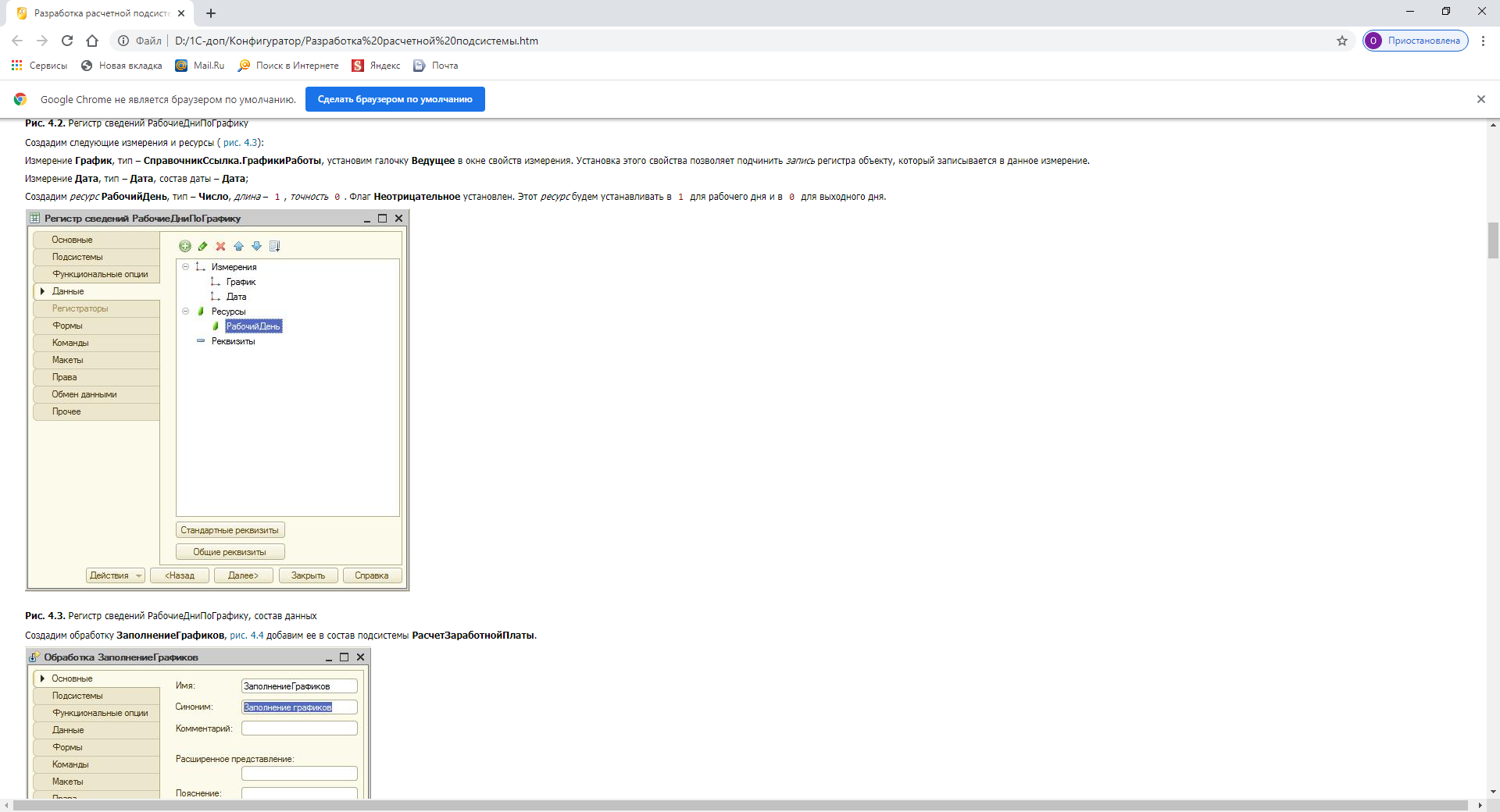
**Рис. 4.2.**Регистр сведений РабочиеДниПоГрафику

Создадим следующие измерения и ресурсы ([рис. 4.3](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=2#image.4.3)):

Измерение **График**, тип – **СправочникСсылка.ГрафикиРаботы**, установим галочку **Ведущее** в окне свойств измерения. Установка этого свойства позволяет подчинить *запись* регистра объекту, который записывается в данное измерение.

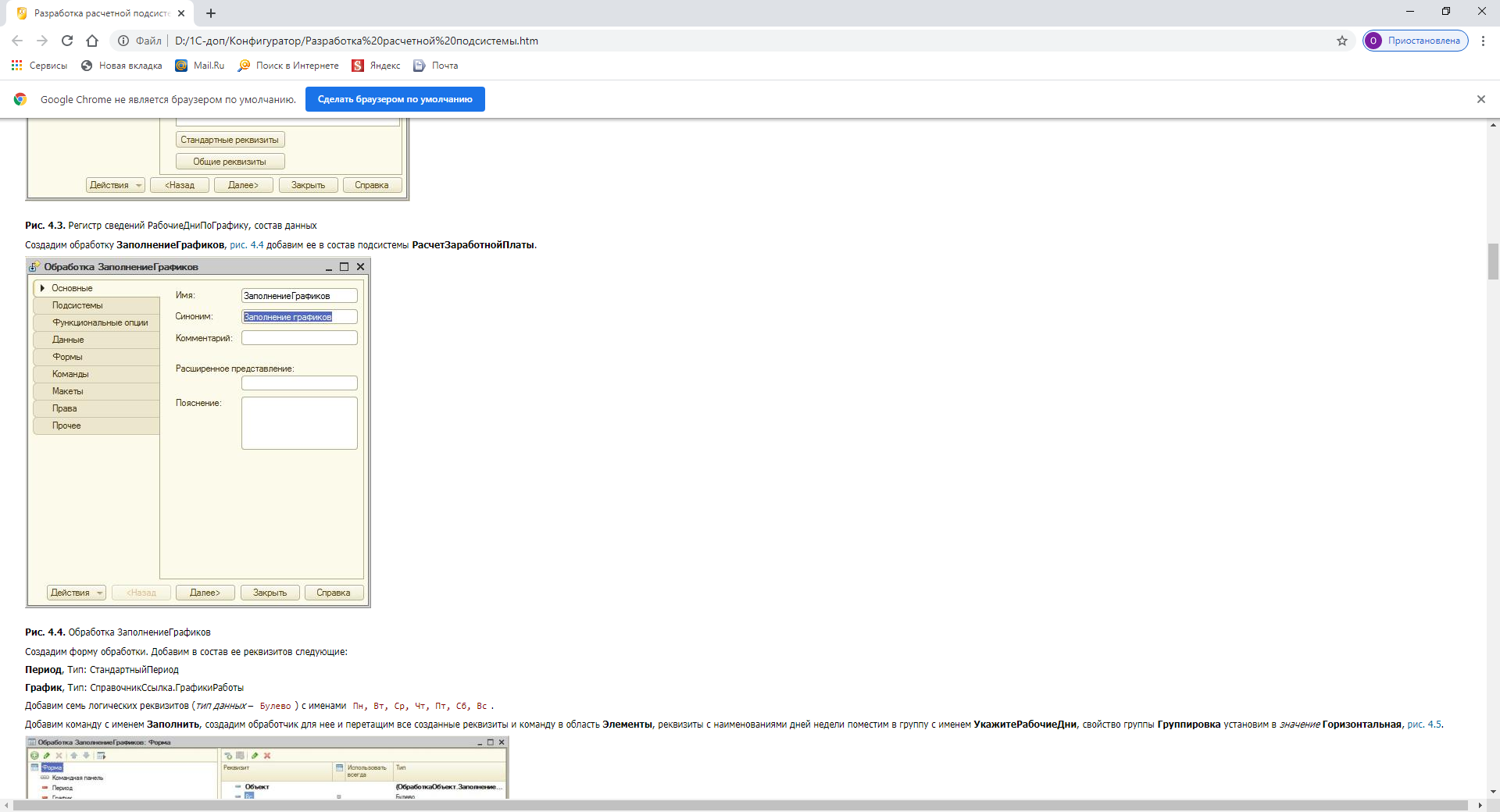
Измерение **Дата**, тип – **Дата**, состав даты – **Дата**;

Создадим *ресурс* **РабочийДень**, тип – **Число**, *длина* – 1, *точность* 0. Флаг **Неотрицательное** установлен. Этот *ресурс* будем устанавливать в 1 для рабочего дня и в 0 для выходного дня.



**Рис. 4.3.**Регистр сведений РабочиеДниПоГрафику, состав данных

Создадим обработку **ЗаполнениеГрафиков**, [рис. 4.4](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=2#image.4.4) добавим ее в состав подсистемы **РасчетЗаработнойПлаты**.

  
**Рис. 4.4.**Обработка ЗаполнениеГрафиков

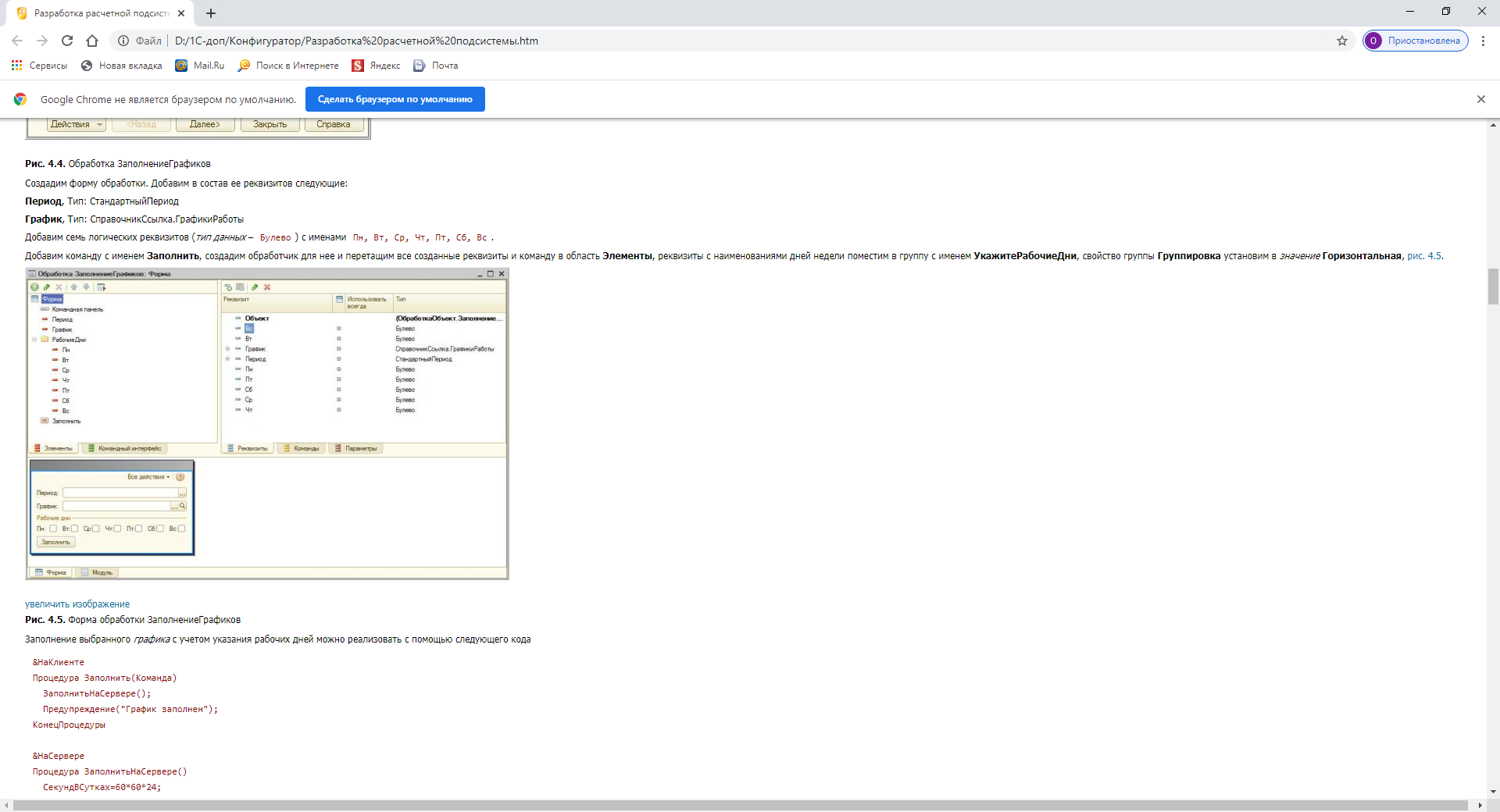
Создадим форму обработки. Добавим в состав ее реквизитов следующие:

**Период**, Тип: СтандартныйПериод

**График**, Тип: СправочникСсылка.ГрафикиРаботы

Добавим семь логических реквизитов (*тип данных* – Булево) с именами Пн, Вт, Ср, Чт, Пт, Сб, Вс.

Добавим команду с именем **Заполнить**, создадим обработчик для нее и перетащим все созданные реквизиты и команду в область **Элементы**, реквизиты с наименованиями дней недели поместим в группу с именем **УкажитеРабочиеДни**, свойство группы **Группировка** установим в *значение* **Горизонтальная**, [рис. 4.5](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=3#image.4.5).

  
  
**Рис. 4.5.**Форма обработки ЗаполнениеГрафиков

Заполнение выбранного *графика* с учетом указания рабочих дней можно реализовать с помощью следующего кода

&НаКлиенте

Процедура Заполнить(Команда)

ЗаполнитьНаСервере();

Предупреждение("График заполнен");

КонецПроцедуры

&НаСервере

Процедура ЗаполнитьНаСервере()

СекундВСутках=60\*60\*24;

МассивРабочихДней=Новый Массив(7);

МассивРабочихДней[0]=Пн;

МассивРабочихДней[1]=Вт;

МассивРабочихДней[2]=Ср;

МассивРабочихДней[3]=Чт;

МассивРабочихДней[4]=Пт;

МассивРабочихДней[5]=Сб;

МассивРабочихДней[6]=Вс;

КоличествоДней = (НачалоДня(Период.ДатаОкончания)

- НачалоДня(Период.ДатаНачала)) / СекундВСутках;

Для НомерДня=0 По КоличествоДней Цикл

ТекущаяДата=Период.ДатаНачала+СекундВСутках\*НомерДня;

Запись=РегистрыСведений.РабочиеДниПоГрафику.СоздатьМенеджерЗаписи();

Запись.График=График;

Запись.Дата=ТекущаяДата;

Запись.РабочийДень=?(МассивРабочихДней[ДеньНедели(ТекущаяДата)-1],1,0);

Запись.Записать();

КонецЦикла;

КонецПроцедуры

Процедура команды – это клиентская процедура, а нам нужно работать на сервере, так как мы собираемся выполнять *операции* с регистром сведений, а именно – создавать его элементы.

Мы вызываем из клиентской процедуры Заполнить() процедуру ЗаполнитьНаСервере().

В серверной процедуре мы сначала записываем в переменную СекундВСутках количество секунд в сутках – это *значение* пригодится нам в дальнейших вычислениях – при работе со значениями, представляющими собой дату, приходится оперировать именно секундами.

Далее, мы создаем новый *массив*, который называем МассивРабочихДней – этот *массив* будет содержать значения, установленные в логических реквизитах, размещенных на форме, то есть – либо *значение* Ложь, либо – *значение* Истина. В массиве семь элементов, по числу дней недели, *нумерация* элементов начинается с нуля.

Мы, в переменную КоличествоРабочихДней вносим количество дней, за которое нужно заполнить *график*. Получаем мы его, вычитая *значение* свойства ДатаНачала из значения ДатаОкончания – эти стандартные свойства получило *поле* Период, когда мы установили его тип как СтандартныйПериод. Полученное *значение* будет выражено в секундах, поэтому мы делим его на ранее найденное количество секунд в сутках.

Цикл Для оперирует счетчиком *цикла* НомерДня, который начинается с 0, что будет представлять дату начала и изменяется с шагом 1 до ранее найденного количества дней.

В цикле мы находим текущую дату, прибавляя к начальной дате *произведение* номера дня и количества секунд в сутках, после чего создаем новый *менеджер* записи для регистра сведений РабочиеДниПоГрафику, заполняем данные записи регистра.

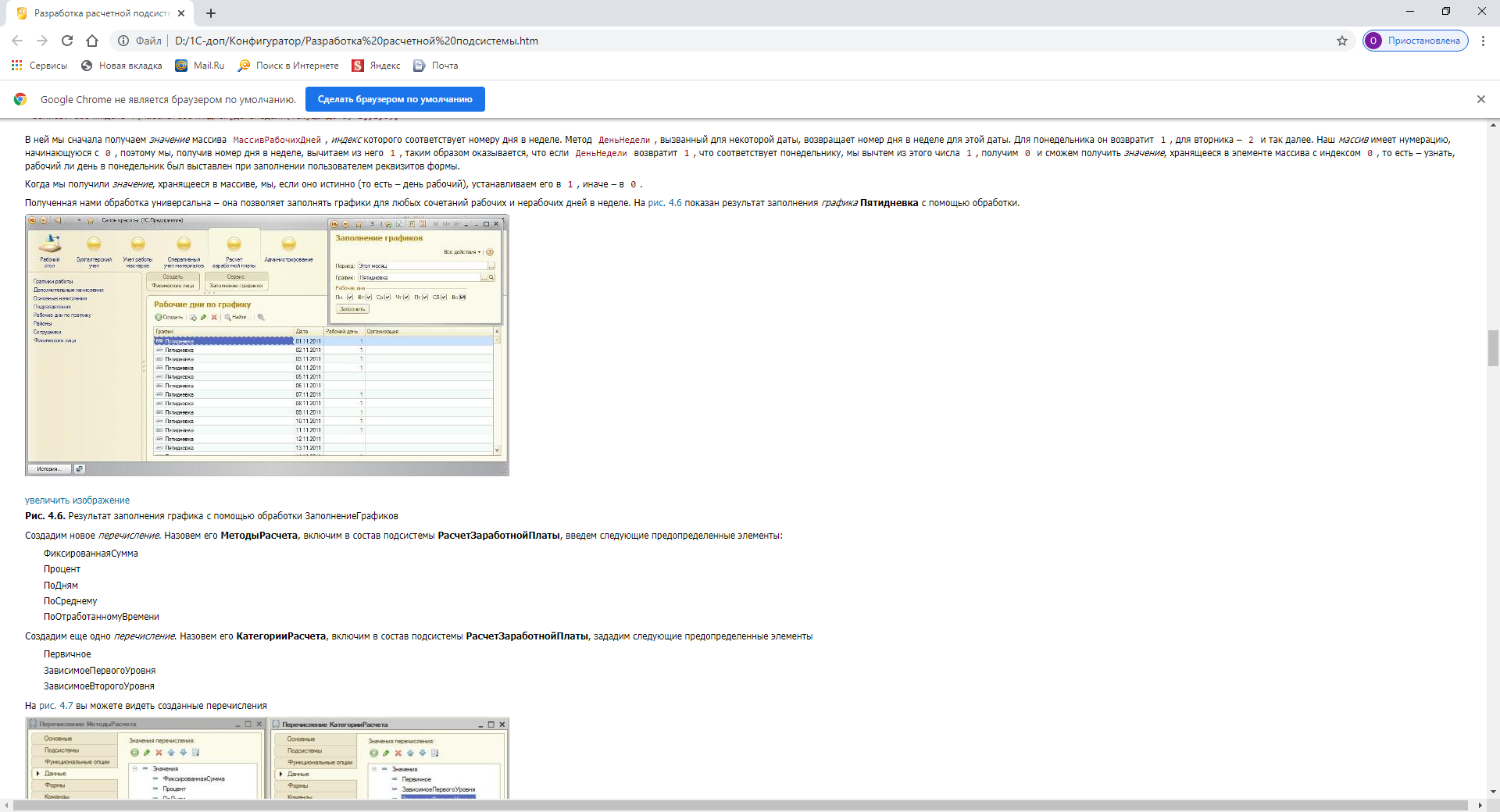
При заполнении ресурса Рабочий день мы используем следующую конструкцию:

Запись.РабочийДень=?(МассивРабочихДней[ДеньНедели(ТекущаяДата)-1],1,0);

В ней мы сначала получаем *значение* массива МассивРабочихДней, *индекс* которого соответствует номеру дня в неделе. Метод ДеньНедели, вызванный для некоторой даты, возвращает номер дня в неделе для этой даты. Для понедельника он возвратит 1, для вторника – 2 и так далее. Наш *массив* имеет нумерацию, начинающуюся с 0, поэтому мы, получив номер дня в неделе, вычитаем из него 1, таким образом оказывается, что если ДеньНедели возвратит 1, что соответствует понедельнику, мы вычтем из этого числа 1, получим 0 и сможем получить *значение*, хранящееся в элементе массива с индексом 0, то есть – узнать, рабочий ли день в понедельник был выставлен при заполнении пользователем реквизитов формы.

Когда мы получили *значение*, хранящееся в массиве, мы, если оно истинно (то есть – день рабочий), устанавливаем его в 1, иначе – в 0.

Полученная нами обработка универсальна – она позволяет заполнять графики для любых сочетаний рабочих и нерабочих дней в неделе. На [рис. 4.6](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=3#image.4.6) показан результат заполнения *графика* **Пятидневка** с помощью обработки.

  
  
**Рис. 4.6.**Результат заполнения графика с помощью обработки ЗаполнениеГрафиков

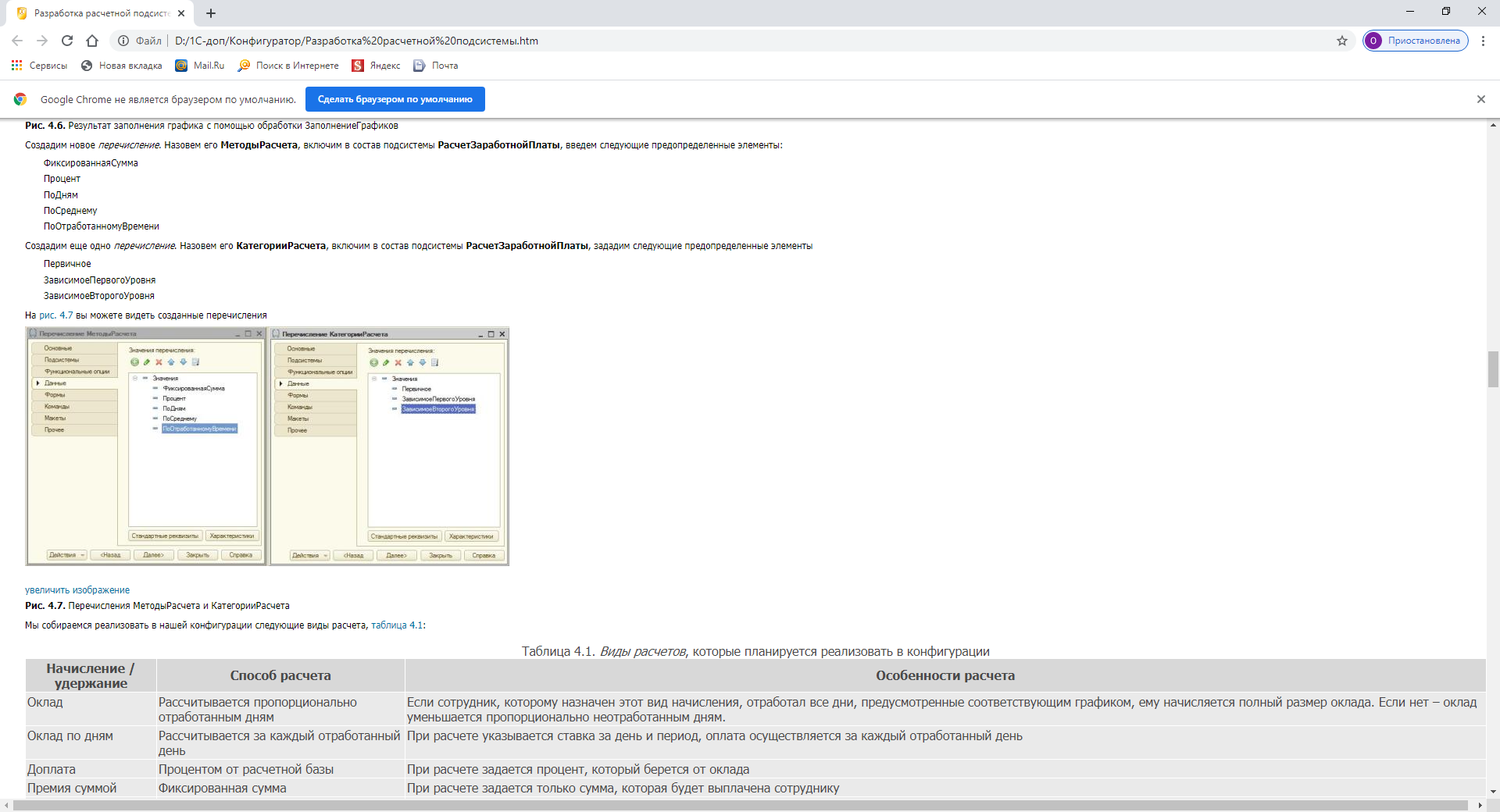
Создадим новое *перечисление*. Назовем его **МетодыРасчета**, включим в состав подсистемы **РасчетЗаработнойПлаты**, введем следующие предопределенные элементы:

* ФиксированнаяСумма
* Процент
* ПоДням
* ПоСреднему
* ПоОтработанномуВремени

Создадим еще одно *перечисление*. Назовем его **КатегорииРасчета**, включим в состав подсистемы **РасчетЗаработнойПлаты**, зададим следующие предопределенные элементы

* Первичное
* ЗависимоеПервогоУровня
* ЗависимоеВторогоУровня

На [рис. 4.7](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=3#image.4.7) вы можете видеть созданные перечисления

  
  
**Рис. 4.7.**Перечисления МетодыРасчета и КатегорииРасчета

Мы собираемся реализовать в нашей конфигурации следующие виды расчета, [таблица 4.1](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=4#table.4.1):

|  |  |  |
| --- | --- | --- |
| Таблица 4.1. *Виды расчетов*, которые планируется реализовать в конфигурации | | |
| **Начисление / удержание** | **Способ расчета** | **Особенности расчета** |
| Оклад | Рассчитывается пропорционально отработанным дням | Если сотрудник, которому назначен этот вид начисления, отработал все дни, предусмотренные соответствующим графиком, ему начисляется полный размер оклада. Если нет – оклад уменьшается пропорционально неотработанным дням. |
| Оклад по дням | Рассчитывается за каждый отработанный день | При расчете указывается ставка за день и период, оплата осуществляется за каждый отработанный день |
| Доплата | Процентом от расчетной базы | При расчете задается процент, который берется от оклада |
| Премия суммой | Фиксированная сумма | При расчете задается только сумма, которая будет выплачена сотруднику |
| Премия | Процентом от расчетной базы | При расчете задается процент, который берется от оклада, который начислен за текущий месяц, и от доплаты за текущий месяц |
| Удержание | Фиксированная сумма | Задается фиксированная сумма, которая будет удержана с сотрудника |
| Прогул | Нулевая сумма | Если у сотрудника был прогул, этот факт фиксируется вводом информации о прогуле. При расчете заработной платы оклад не платится за те дни, в которые у сотрудника был прогул. |
| Отпуск | По среднедневному заработку | Рассчитывается по количеству дней отпуска, умноженному на среднедневной заработок, полученный от оклада и от оклада по дням |

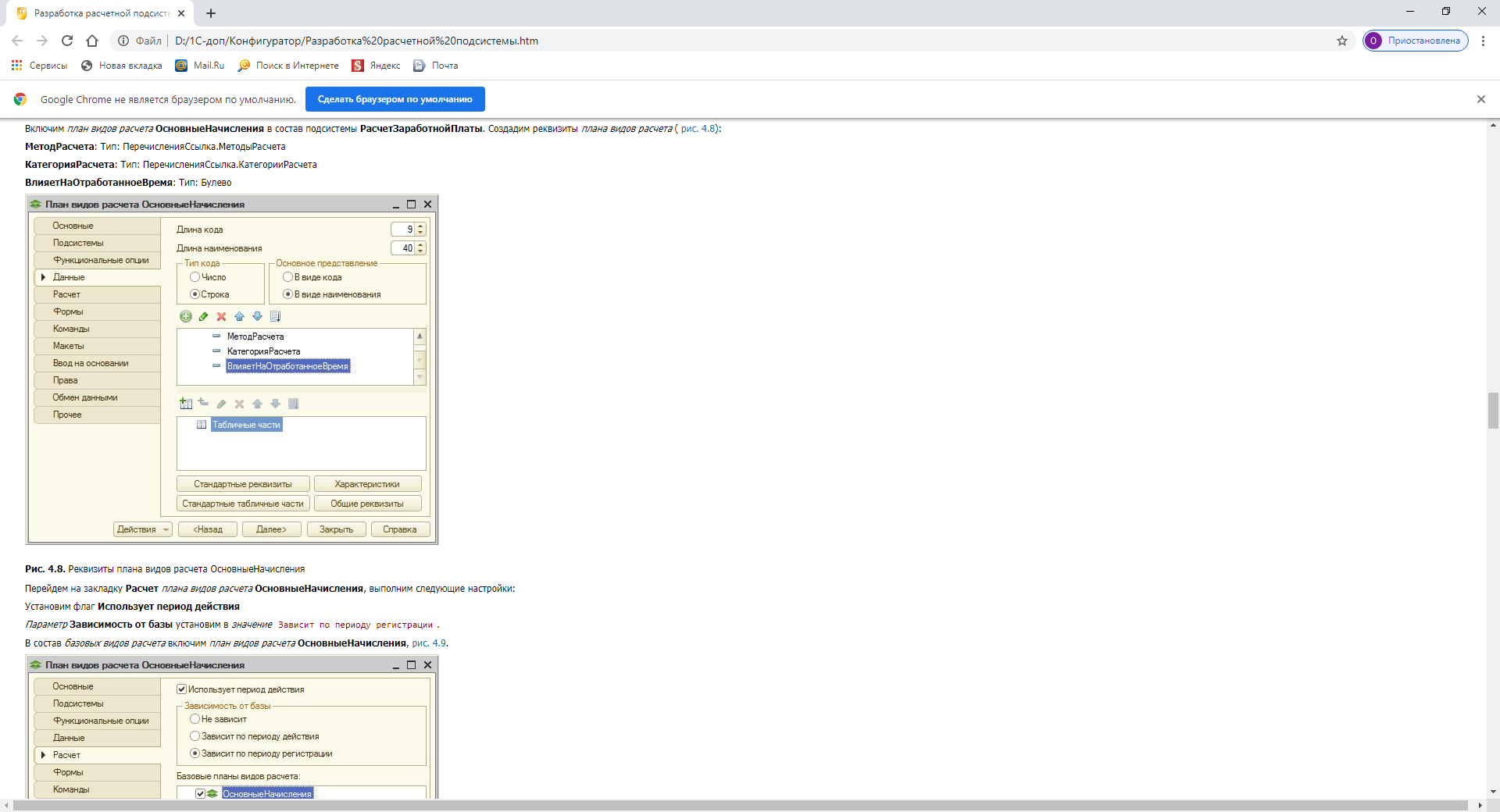
Для задания их в системе создадим два *плана видов расчета*: **ОсновныеНачисления** и **ДополнительныеНачисления**.

Включим *план видов расчета* **ОсновныеНачисления** в состав подсистемы **РасчетЗаработнойПлаты**. Создадим реквизиты *плана видов расчета* ([рис. 4.8](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=4#image.4.8)):

**МетодРасчета**: Тип: ПеречисленияСсылка.МетодыРасчета

**КатегорияРасчета**: Тип: ПеречисленияСсылка.КатегорииРасчета

**ВлияетНаОтработанноеВремя**: Тип: Булево

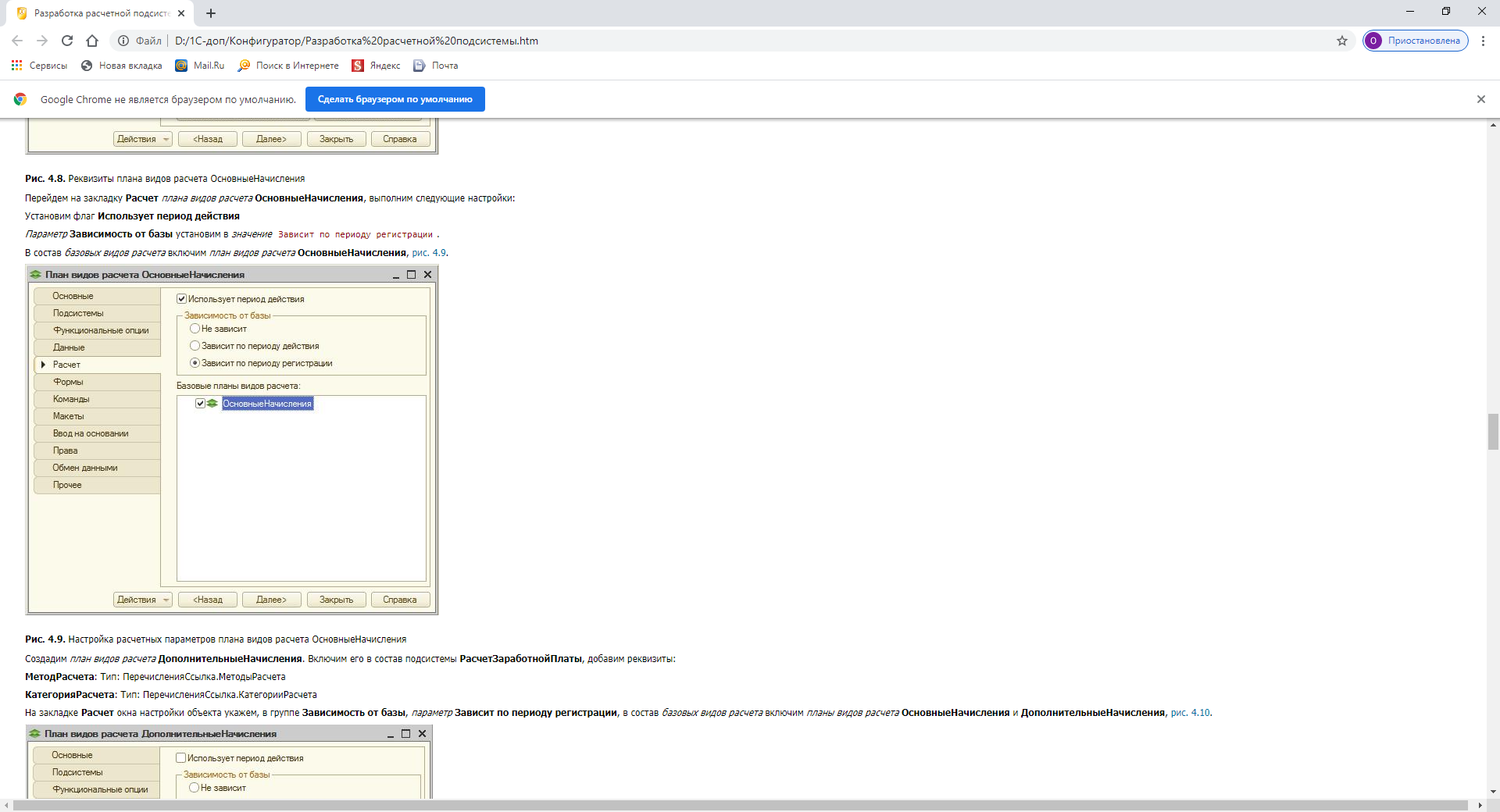
  
**Рис. 4.8.**Реквизиты плана видов расчета ОсновныеНачисления

Перейдем на закладку **Расчет** *плана видов расчета* **ОсновныеНачисления**, выполним следующие настройки:

Установим флаг **Использует период действия**

*Параметр* **Зависимость от базы** установим в *значение* Зависит по периоду регистрации.

В состав *базовых видов расчета* включим *план видов расчета* **ОсновныеНачисления**, [рис. 4.9](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=4#image.4.9).



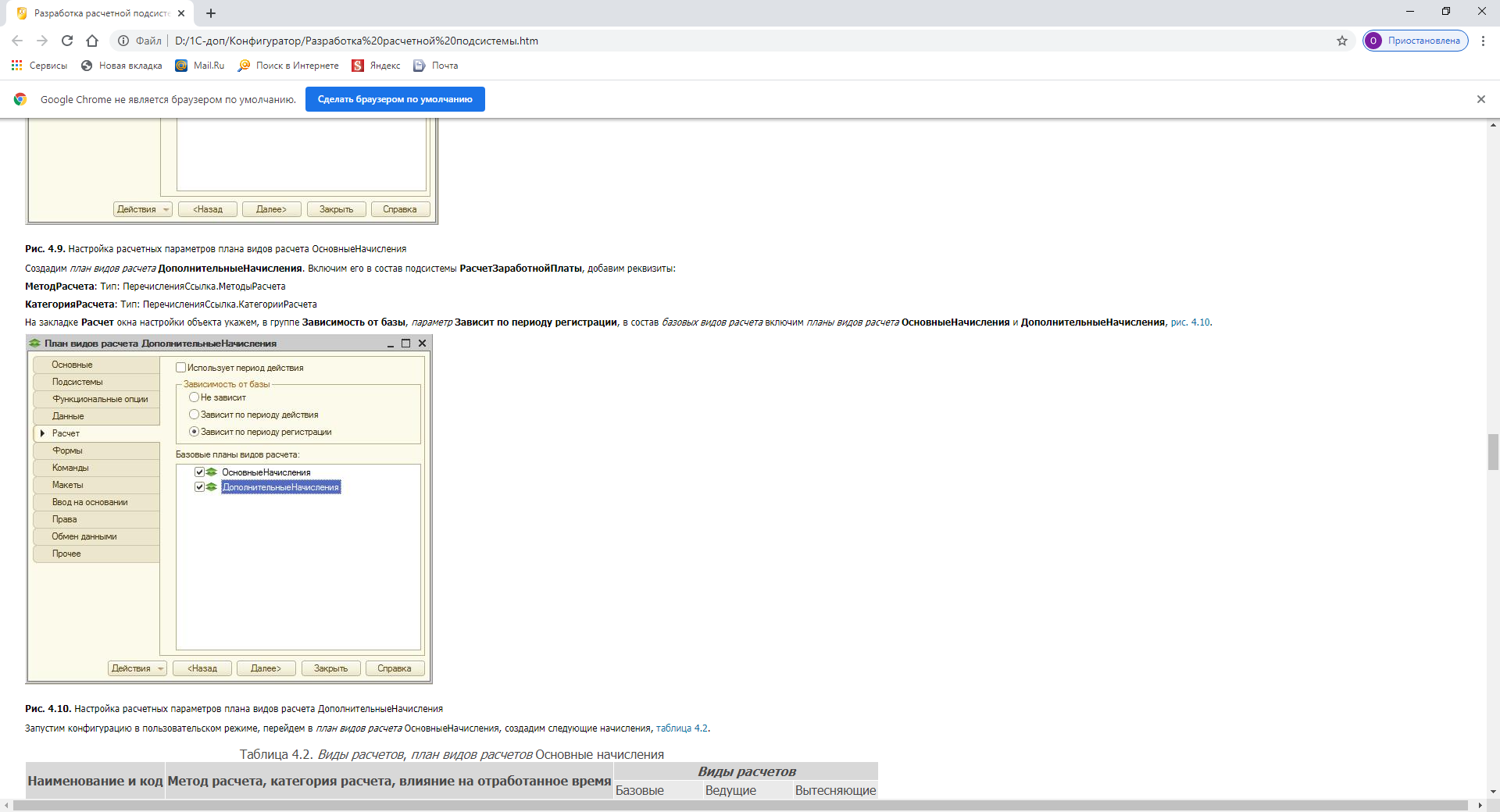
**Рис. 4.9.**Настройка расчетных параметров плана видов расчета ОсновныеНачисления

Создадим *план видов расчета* **ДополнительныеНачисления**. Включим его в состав подсистемы **РасчетЗаработнойПлаты**, добавим реквизиты:

**МетодРасчета**: Тип: ПеречисленияСсылка.МетодыРасчета

**КатегорияРасчета**: Тип: ПеречисленияСсылка.КатегорииРасчета

На закладке **Расчет** окна настройки объекта укажем, в группе **Зависимость от базы**, *параметр* **Зависит по периоду регистрации**, в состав *базовых видов расчета* включим *планы видов расчета* **ОсновныеНачисления** и **ДополнительныеНачисления**, [рис. 4.10](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=4#image.4.10).



**Рис. 4.10.**Настройка расчетных параметров плана видов расчета ДополнительныеНачисления

Запустим конфигурацию в пользовательском режиме, перейдем в *план видов расчета* ОсновныеНачисления, создадим следующие начисления, [таблица 4.2](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=5#table.4.2).

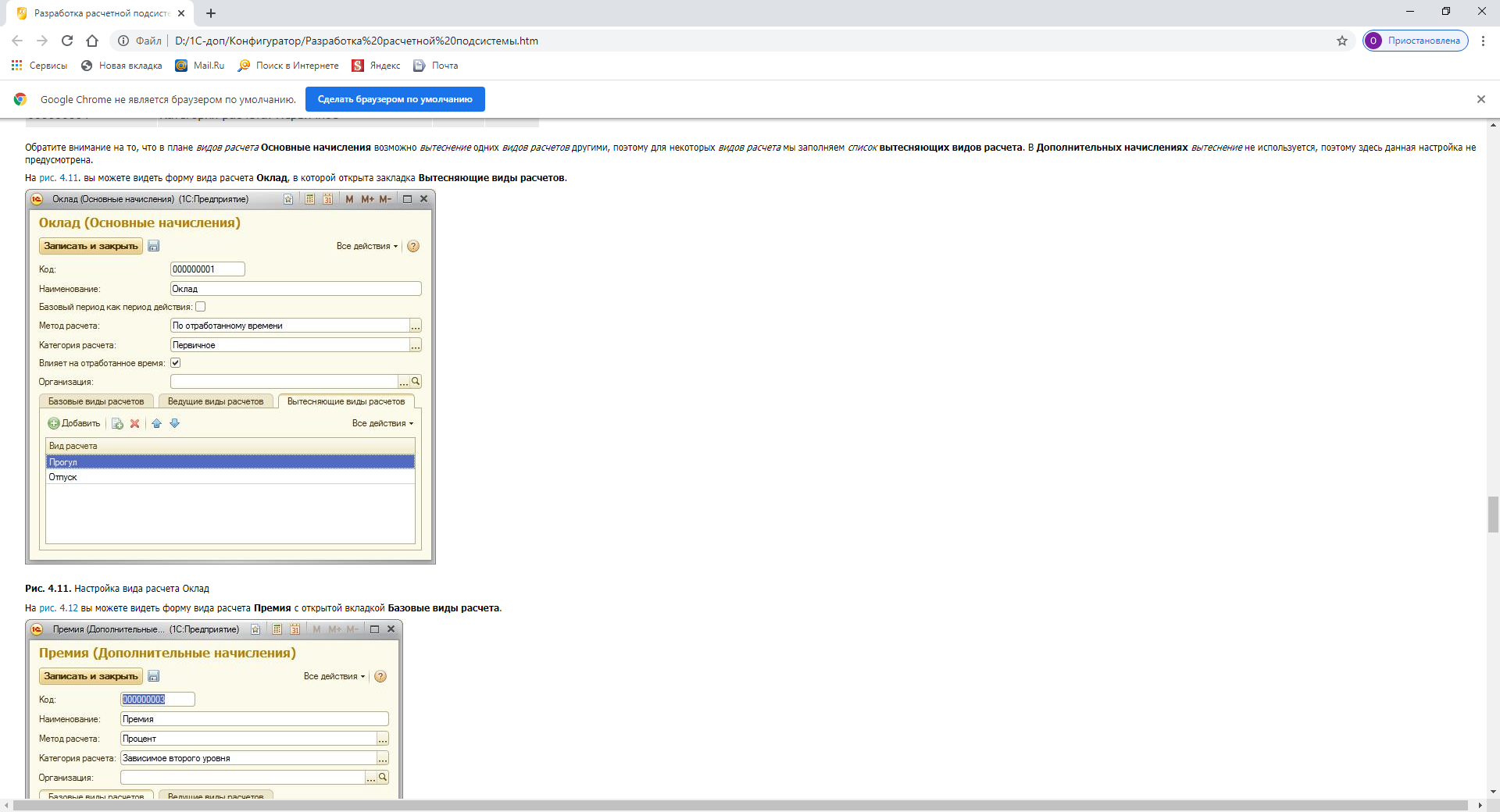
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица 4.2. *Виды расчетов*, *план видов расчетов* Основные начисления | | | | |
| **Наименование и код** | **Метод расчета, категория расчета, влияние на отработанное время** | ***Виды расчетов*** | | |
| Базовые | Ведущие | Вытесняющие |
| Оклад  000000001 | Метод расчета: По отработанному времени  Категория расчета: Первичное  Влияет на отработанное время: Установлено |  |  | Прогул  Отпуск |
| Оклад по дням  000000002 | Метод расчета: По дням  Категория расчета: Первичное  Влияет на отработанное время: Установлено |  |  | Прогул  Отпуск |
| Прогул  000000003 | Метод расчета: Фиксированная сумма  Категория расчета: Первичное  Влияет на отработанное время: Не установлено |  |  |  |
| Отпуск  000000004 | Метод расчета: По среднему  Категория расчета: Зависимое первого уровня  Влияет на отработанное время: Не установлено | Оклад  Оклад по дням | Оклад  Оклад по дням |  |

Перейдем в *план видов расчета* **Дополнительные начисления**, создадим следующие *виды расчетов*, [таблица 4.3](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=5#table.4.3).

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 4.3. *Виды расчетов*, *план видов расчетов* Дополнительные начисления | | | |
| **Наименование, код** | **Метод расчета, категория расчета** | ***Виды расчетов*** | |
| Базовые | Ведущие |
| Доплата  000000001 | Метод расчета: Процентом  Категория расчета: Зависимое первого уровня | Оклад | Оклад |
| Премия суммой  000000002 | Метод расчета: Фиксированная сумма  Категория расчета: Первичное |  |  |
| Премия  000000003 | Метод расчета: Процентом  Категория расчета: Зависимое второго уровня | Оклад  Доплата | Оклад  Доплата |
| Удержание  000000004 | Метод расчета: Фиксированная сумма  Категория расчета: Первичное |  |  |

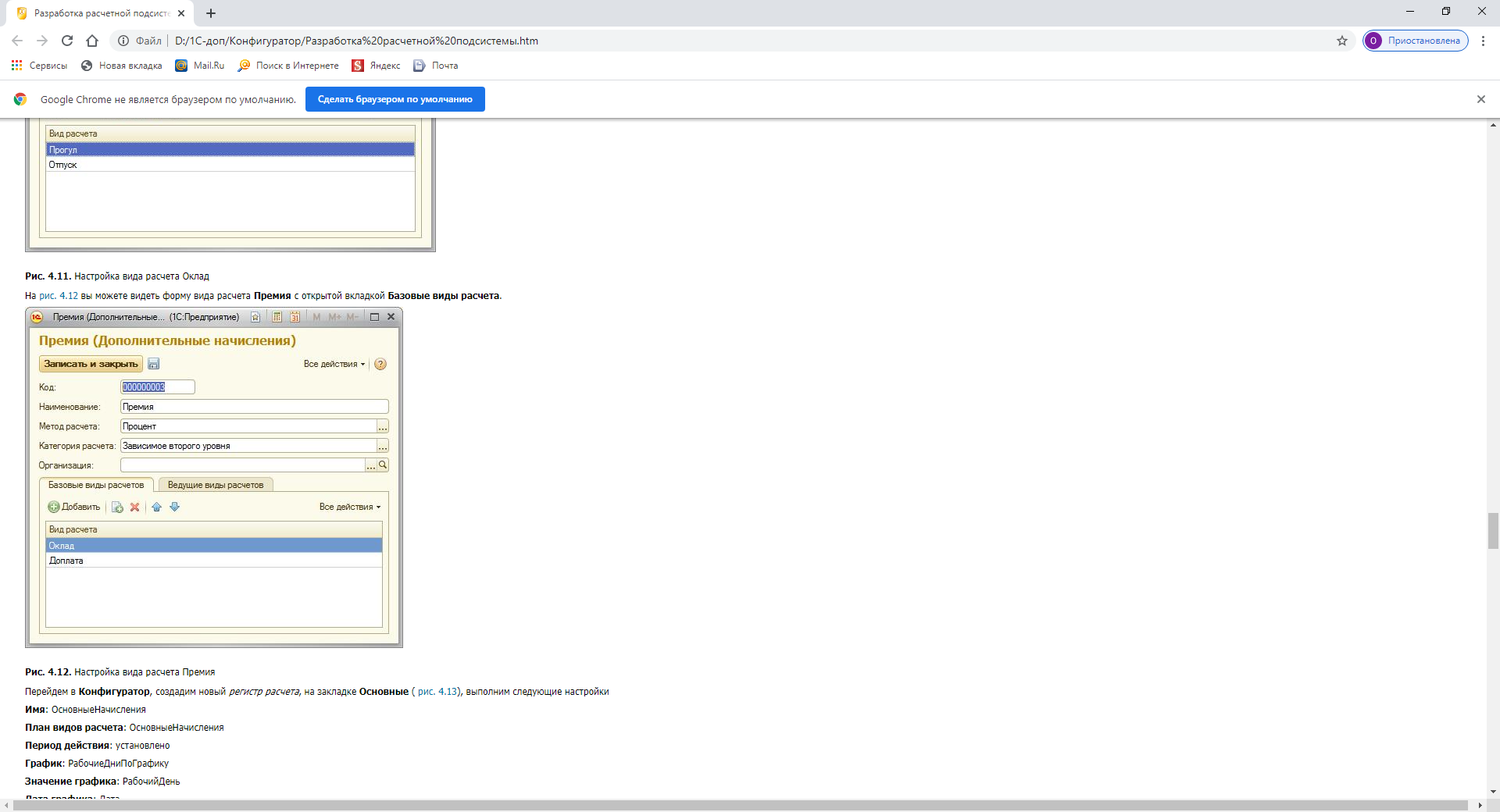
Обратите внимание на то, что в плане *видов расчета* **Основные начисления** возможно *вытеснение* одних *видов расчетов* другими, поэтому для некоторых *видов расчета* мы заполняем *список* **вытесняющих видов расчета**. В **Дополнительных начислениях** *вытеснение* не используется, поэтому здесь данная настройка не предусмотрена.

На [рис. 4.11](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=5#image.4.11). вы можете видеть форму вида расчета **Оклад**, в которой открыта закладка **Вытесняющие виды расчетов**.



**Рис. 4.11.**Настройка вида расчета Оклад

На [рис. 4.12](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=5#image.4.12) вы можете видеть форму вида расчета **Премия** с открытой вкладкой **Базовые виды расчета**.



**Рис. 4.12.**Настройка вида расчета Премия

Перейдем в **Конфигуратор**, создадим новый *регистр расчета*, на закладке **Основные** ([рис. 4.13](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=5#image.4.13)), выполним следующие настройки

**Имя**: ОсновныеНачисления

**План видов расчета**: ОсновныеНачисления

**Период действия**: установлено

**График**: РабочиеДниПоГрафику

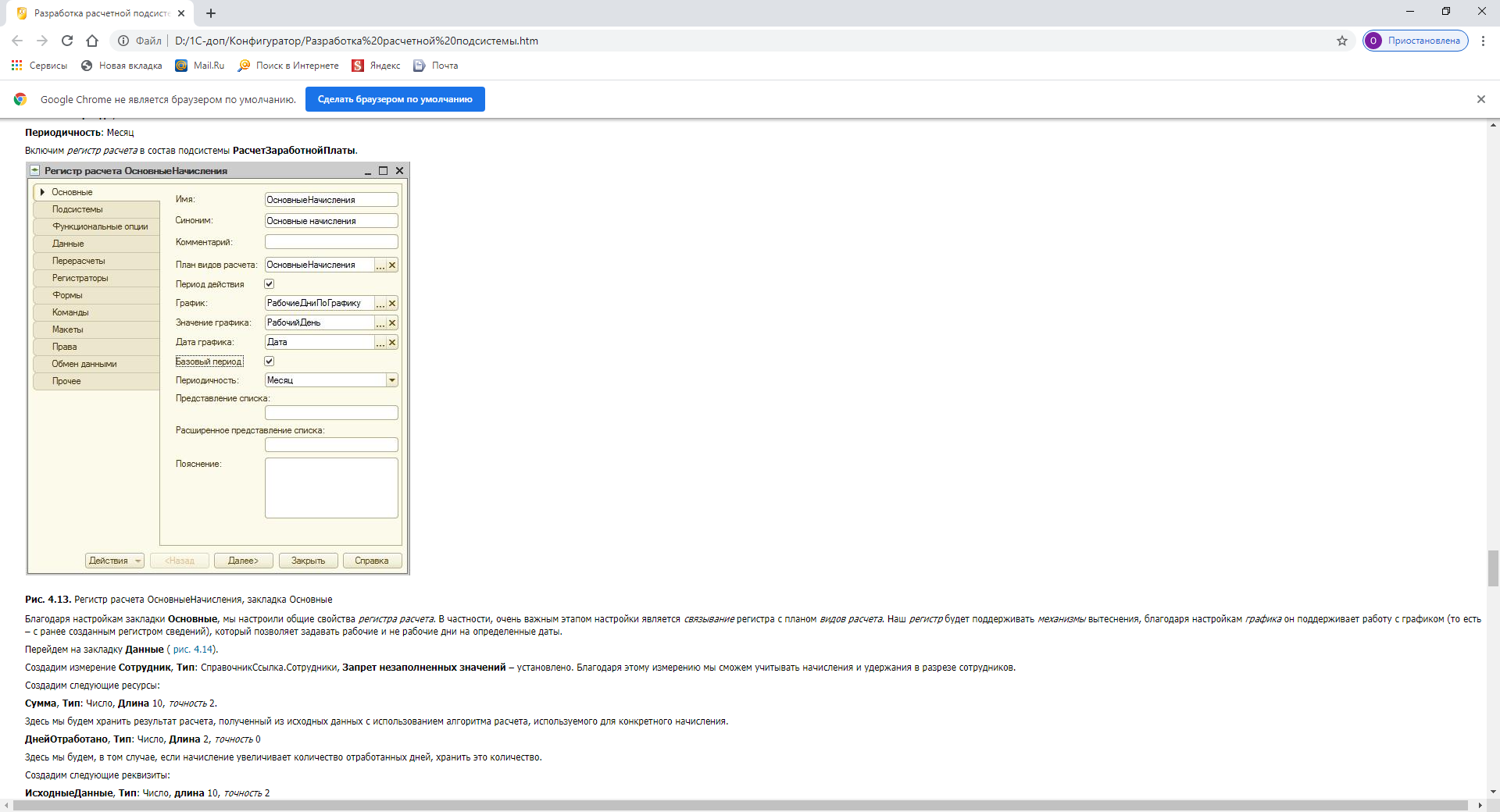
**Значение графика**: РабочийДень

**Дата графика**: Дата

**Базовый период**: установлено

**Периодичность**: Месяц

Включим *регистр расчета* в состав подсистемы **РасчетЗаработнойПлаты**.



**Рис. 4.13.**Регистр расчета ОсновныеНачисления, закладка Основные

Благодаря настройкам закладки **Основные**, мы настроили общие свойства *регистра расчета*. В частности, очень важным этапом настройки является *связывание* регистра с планом *видов расчета*. Наш *регистр* будет поддерживать *механизмы* вытеснения, благодаря настройкам *графика* он поддерживает работу с графиком (то есть – с ранее созданным регистром сведений), который позволяет задавать рабочие и не рабочие дни на определенные даты.

Перейдем на закладку **Данные** ([рис. 4.14](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=6#image.4.14)).

Создадим измерение **Сотрудник**, **Тип**: СправочникСсылка.Сотрудники, **Запрет незаполненных значений** – установлено. Благодаря этому измерению мы сможем учитывать начисления и удержания в разрезе сотрудников.

Создадим следующие ресурсы:

**Сумма**, **Тип**: Число, **Длина** 10, *точность* 2.

Здесь мы будем хранить результат расчета, полученный из исходных данных с использованием алгоритма расчета, используемого для конкретного начисления.

**ДнейОтработано**, **Тип**: Число, **Длина** 2, *точность* 0

Здесь мы будем, в том случае, если начисление увеличивает количество отработанных дней, хранить это количество.

Создадим следующие реквизиты:

**ИсходныеДанные**, **Тип**: Число, **длина** 10, *точность* 2

В этом реквизите мы будем хранить исходные данные для расчета вида начисления (суммы, процентные показатели).

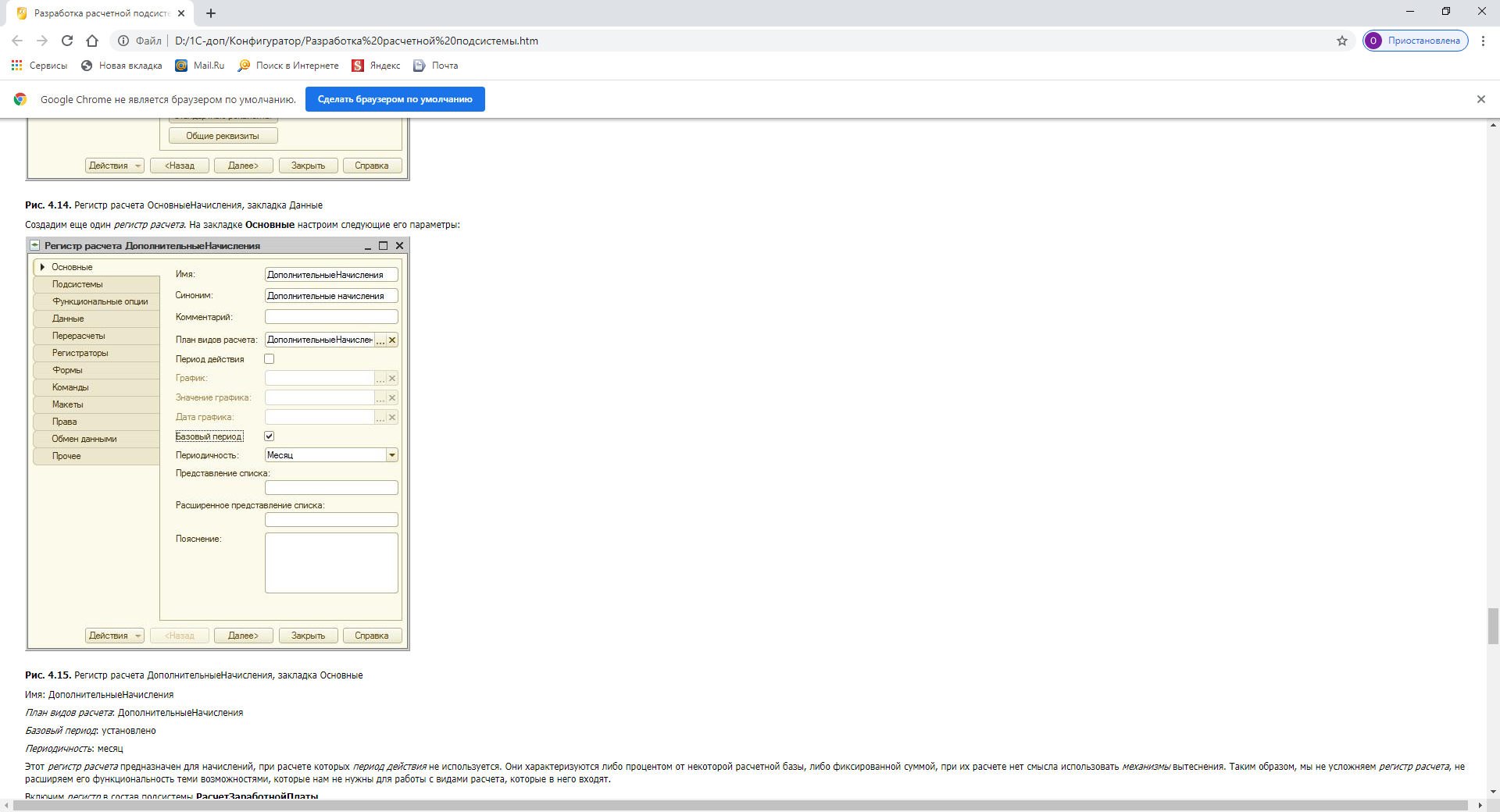
**График**: **Тип**: СправочникСсылка.ГрафикиРаботы, **Связь с графиком** – *График* (измерение регистра сведений)

Здесь мы будем хранить ссылку на *график*, используемый с начислением в том случае, если это имеет смысл.



**Рис. 4.14.**Регистр расчета ОсновныеНачисления, закладка Данные

Создадим еще один *регистр расчета*. На закладке **Основные** настроим следующие его параметры:

  
**Рис. 4.15.**Регистр расчета ДополнительныеНачисления, закладка Основные

Имя: ДополнительныеНачисления

*План видов расчета*: ДополнительныеНачисления

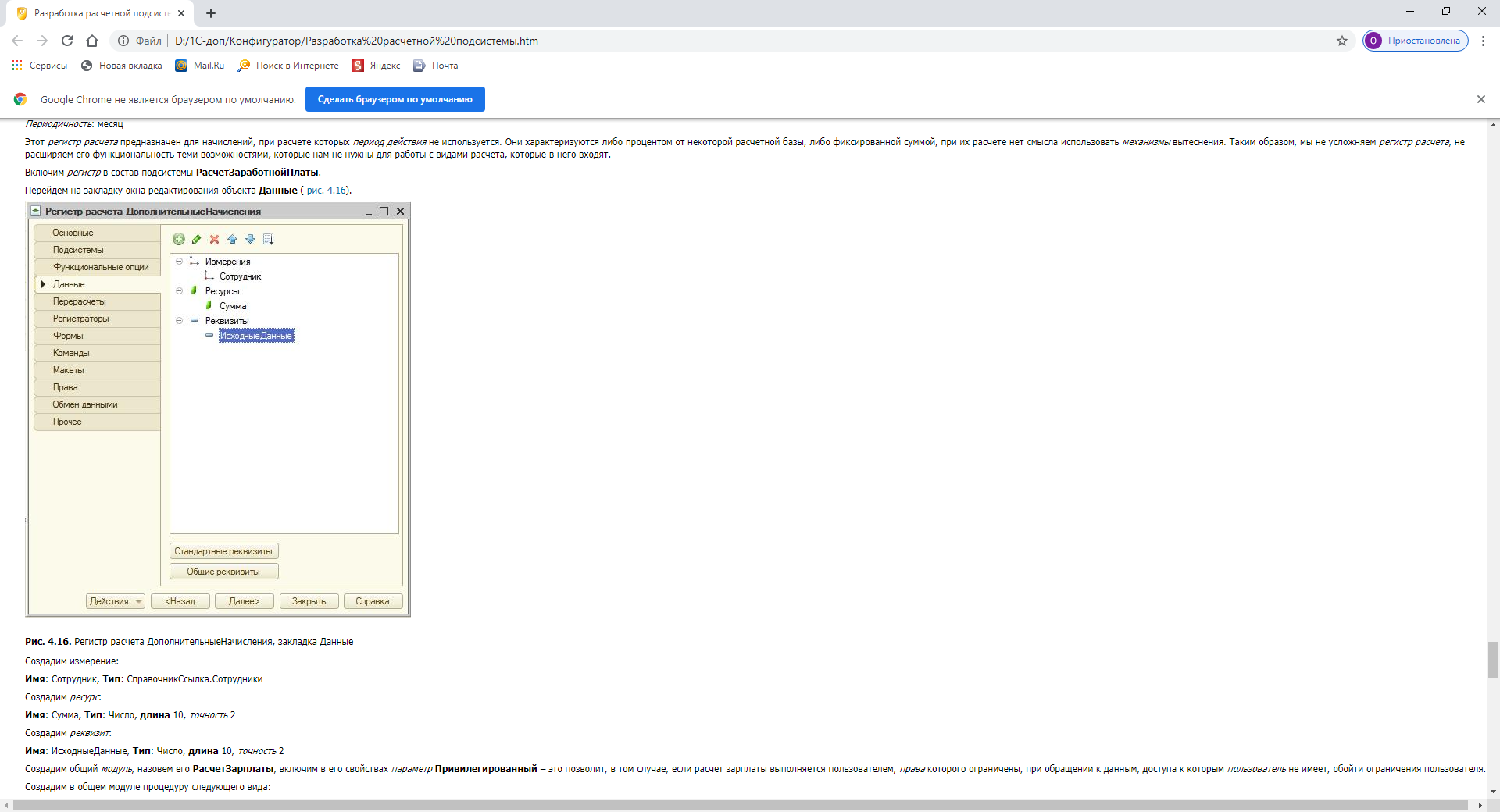
*Базовый период*: установлено

*Периодичность*: месяц

Этот *регистр расчета* предназначен для начислений, при расчете которых *период действия* не используется. Они характеризуются либо процентом от некоторой расчетной базы, либо фиксированной суммой, при их расчете нет смысла использовать *механизмы* вытеснения. Таким образом, мы не усложняем *регистр расчета*, не расширяем его функциональность теми возможностями, которые нам не нужны для работы с видами расчета, которые в него входят.

Включим *регистр* в состав подсистемы **РасчетЗаработнойПлаты**.

Перейдем на закладку окна редактирования объекта **Данные** ([рис. 4.16](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=6#image.4.16)).

  
**Рис. 4.16.**Регистр расчета ДополнительныеНачисления, закладка Данные

Создадим измерение:

**Имя**: Сотрудник, **Тип**: СправочникСсылка.Сотрудники

Создадим *ресурс*:

**Имя**: Сумма, **Тип**: Число, **длина** 10, *точность* 2

Создадим *реквизит*:

**Имя**: ИсходныеДанные, **Тип**: Число, **длина** 10, *точность* 2

Создадим общий *модуль*, назовем его **РасчетЗарплаты**, включим в его свойствах *параметр* **Привилегированный** – это позволит, в том случае, если расчет зарплаты выполняется пользователем, *права* которого ограничены, при обращении к данным, доступа к которым *пользователь* не имеет, обойти ограничения пользователя.

Создадим в общем модуле процедуру следующего вида:

Процедура РассчитатьЗарплату(Регистратор) Экспорт

КонецПроцедуры

Мы будем вызывать эту процедуру из процедуры обработки проведения документа. Мы будем передавать в процедуру *параметр* Регистратор – ссылку на документ, который сформировал движения в регистре. Отобрав записи регистра по переданному регистратору, в дальнейшем мы реализуем расчет записей.

Теперь займемся документом, который будет использоваться для начисления заработной платы.

Создадим новый документ, назовем его **НачислениеЗарплаты**, укажем, что он входит в подсистему **РасчетЗаработнойПлаты**, укажем, что он проводится по *регистрам расчета* **ОсновныеНачисления** и **ДополнительныеНачисления**. Запретим оперативное *проведение документа*.

На закладке **Данные** добавим табличную часть **ОсновныеНачисления**. Добавим в нее следующие реквизиты:

**Сотрудник**: **Тип**: СправочникСсылка.Сотрудники

**ВидРасчета**: **Тип**: ПланВидовРасчетаСсылка.ОсновныеНачисления

**ИсходныеДанные**: **Тип**: Число, *длина* 10, *точность* 2

**График**: **Тип**: СправочникСсылка.ГрафикиРаботы

**ДатаНачалаПериодаДействия**: **Тип**: Дата, состав даты – Дата

**ДатаОкончанияПериодаДействия**: **Тип**: Дата, состав даты – Дата

**ДатаНачалаБазовогоПериода**: **Тип**: Дата, состав даты – Дата

**ДатаОкончанияБазовогоПериода**: **Тип**: Дата, состав даты – Дата

Добавим табличную часть **ДополнительныеНачисления**, создадим в ней следующие реквизиты:

**Сотрудник**: **Тип**: СправочникСсылка.Сотрудники

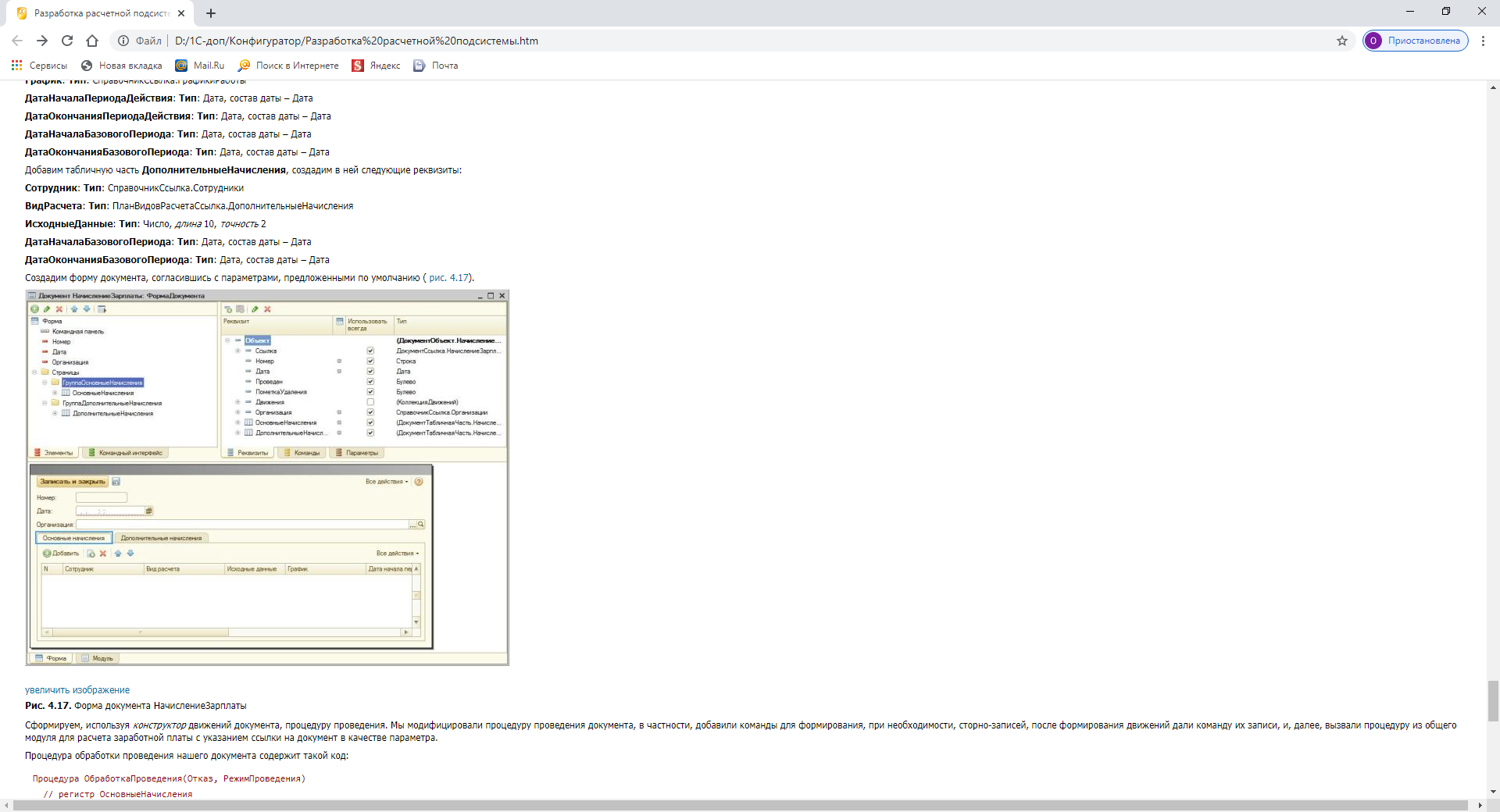
**ВидРасчета**: **Тип**: ПланВидовРасчетаСсылка.ДополнительныеНачисления

**ИсходныеДанные**: **Тип**: Число, *длина* 10, *точность* 2

**ДатаНачалаБазовогоПериода**: **Тип**: Дата, состав даты – Дата

**ДатаОкончанияБазовогоПериода**: **Тип**: Дата, состав даты – Дата

Создадим форму документа, согласившись с параметрами, предложенными по умолчанию ([рис. 4.17](http://www.intuit.ru/studies/courses/2321/621/lecture/13521?page=7#image.4.17)).

  
  
**Рис. 4.17.**Форма документа НачислениеЗарплаты

Сформируем, используя *конструктор* движений документа, процедуру проведения. Мы модифицировали процедуру проведения документа, в частности, добавили команды для формирования, при необходимости, сторно-записей, после формирования движений дали команду их записи, и, далее, вызвали процедуру из общего модуля для расчета заработной платы с указанием ссылки на документ в качестве параметра.

Процедура обработки проведения нашего документа содержит такой код:

Процедура ОбработкаПроведения(Отказ, РежимПроведения)

// регистр ОсновныеНачисления

Движения.ОсновныеНачисления.Записывать = Истина;

Для Каждого ТекСтрокаОсновныеНачисления Из ОсновныеНачисления Цикл

Движение = Движения.ОсновныеНачисления.Добавить();

Движение.Сторно = Ложь;

Движение.ВидРасчета = ТекСтрокаОсновныеНачисления.ВидРасчета;

Движение.ПериодДействияНачало

= ТекСтрокаОсновныеНачисления.ДатаНачалаПериодаДействия;

Движение.ПериодДействияКонец

= ТекСтрокаОсновныеНачисления.ДатаОкончанияПериодаДействия;

Движение.ПериодРегистрации = Дата;

Движение.БазовыйПериодНачало

= ТекСтрокаОсновныеНачисления.ДатаНачалаБазовогоПериода;

Движение.БазовыйПериодКонец

= ТекСтрокаОсновныеНачисления.ДатаОкончанияБазовогоПериода;

Движение.Сотрудник = ТекСтрокаОсновныеНачисления.Сотрудник;

Движение.ИсходныеДанные = ТекСтрокаОсновныеНачисления.ИсходныеДанные;

Движение.График = ТекСтрокаОсновныеНачисления.График;

КонецЦикла;

// регистр ДополнительныеНачисления

Движения.ДополнительныеНачисления.Записывать = Истина;

Для Каждого ТекСтрокаДополнительныеНачисления Из ДополнительныеНачисления Цикл

Движение = Движения.ДополнительныеНачисления.Добавить();

Движение.Сторно = Ложь;

Движение.ВидРасчета = ТекСтрокаДополнительныеНачисления.ВидРасчета;

Движение.ПериодРегистрации = Дата;

Движение.БазовыйПериодНачало

= ТекСтрокаДополнительныеНачисления.ДатаНачалаБазовогоПериода;

Движение.БазовыйПериодКонец

= ТекСтрокаДополнительныеНачисления.ДатаОкончанияБазовогоПериода;

Движение.Сотрудник = ТекСтрокаДополнительныеНачисления.Сотрудник;

Движение.ИсходныеДанные = ТекСтрокаДополнительныеНачисления.ИсходныеДанные;

КонецЦикла;

// Сторно-записи

Дополнения=Движения.ОсновныеНачисления.ПолучитьДополнение();

Для Каждого Дополнение из Дополнения Цикл

Движение=Движения.ОсновныеНачисления.Добавить();

Движение.Сторно=Истина;

Движение.Сотрудник=Дополнение.Сотрудник;

Движение.ВидРасчета=Дополнение.ВидРасчета;

Движение.БазовыйПериодНачало=Дополнение.БазовыйПериодНачало;

Движение.БазовыйПериодКонец=Дополнение.БазовыйПериодКонец;

Движение.График=Дополнение.График;

Движение.ИсходныеДанные=Дополнение.ИсходныеДанные;

Движение.ПериодРегистрации=Дополнение.ПериодРегистрацииСторно;

Движение.ПериодДействияНачало=Дополнение.ПериодДействияНачалоСторно;

Движение.ПериодДействияКонец=Дополнение.ПериодДействияКонецСторно;

КонецЦикла;

Движения.Записать();

РасчетЗарплаты.РассчитатьЗарплату(Ссылка);

КонецПроцедуры

Формирование движений по *регистрам расчета* в циклах перебора табличных частей документа выполняются вполне стандартно.

Обратите внимание на команду:

Дополнения=Движения.ОсновныеНачисления.ПолучитьДополнение();

Метод набора записей *регистра расчета* ПолучитьДополнение() позволяет получить дополнительные данные, с помощью которых можно выполнить сторнирование записей предыдущих периодов при вводе в текущем периоде записей, влияющих на записи предыдущих периодов.

Например, в октябре 2011 года сотруднику был задан *вид расчета* Оклад, а в ноябре 2011 выяснилось, что часть оклада начислена неправомерно – лишь в ноябре стало известно, что несколько дней в предыдущем месяце сотрудник прогулял. В ноябре ему вводится *вид расчета* Прогул, в котором указаны пропущенные дни – Прогул является *вытесняющим видом расчета* для Оклада. За эти дни ему не должен начисляться оклад, но записи предыдущего месяца менять нельзя, например, с них уже начислены налоги. При вводе данных о прогуле система, на основе механизма вытеснения, может "предложить" сформировать сторно-*запись*, которая отразит в текущем периоде данные (*вычет*, в конечном счете), которые приведут, в итоге, к правильным взаиморасчетам организации и сотрудника. Например, если в предыдущем периоде ему было переплачено 1500 рублей, то, в соответствии с идеологией сторнирования, в текущем периоде у него вычтут те же 1500 рублей, не затрагивая, при этом, записи, сформированные в предыдущем периоде.

После того, как записи, и, при необходимости, сторно-записи сформированы, записываем их и вызываем процедуру РассчитатьЗарплату() из общего модуля РасчетЗарплаты.

Расчетом записей мы займемся в следующей работе.