# Теория к практическому занятию №14

# Задачи на работу с анимацией на CSS

**Блок 1.animation-name**

Свойство **animation-name** задает имя анимации (или анимаций), которая будет применяться к данному элементу. Имя анимации должно быть такое же, какое задано при определении ключевых кадров @keyframes.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-name: имя анимации;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-name:имя1, имя2;

}

Значение по умолчанию: **нет (?)**.

**Примеры**

Внимание! Кроссбраузерные приставки для анимации опущены для краткости, но они нужны!.

**Пример 1.**

В данном примере анимация будет проигрываться в течении 3-х секунд:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-name: move;

animation-duration:3s;

animation-iteration-count: infinite;

animation-timing-function:linear;

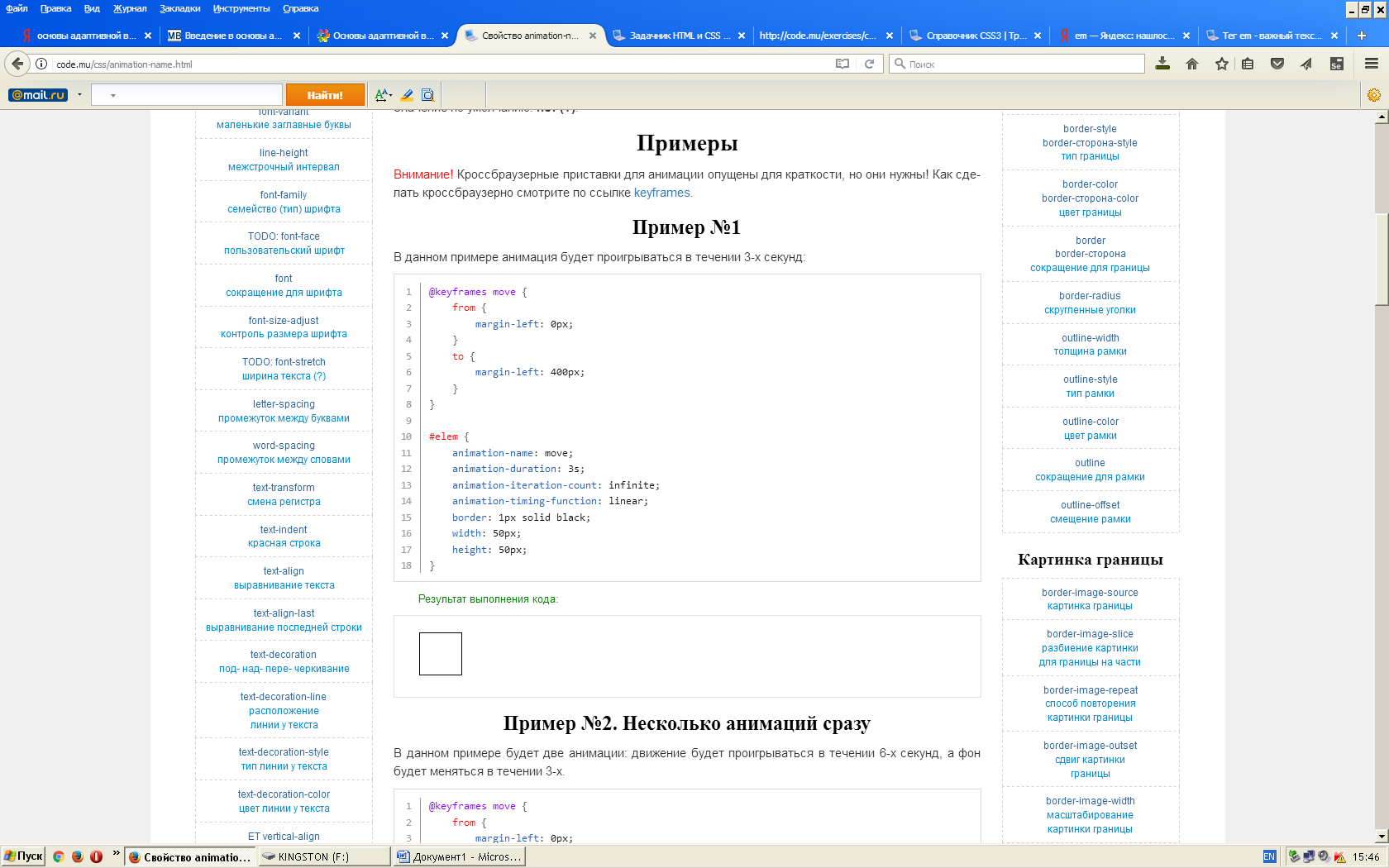
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 2. Несколько анимаций сразу**

В данном примере будет две анимации: движение будет проигрываться в течении 6-х секунд, а фон будет меняться в течении 3-х.

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

@keyframesbg{

from{

background-color:red;

}

to{

background-color:white;

}

}

#elem{

animation-name: move, bg;

animation-duration:6s, 3s;

animation-iteration-count: infinite, infinite;

animation-timing-function: linear, ease;

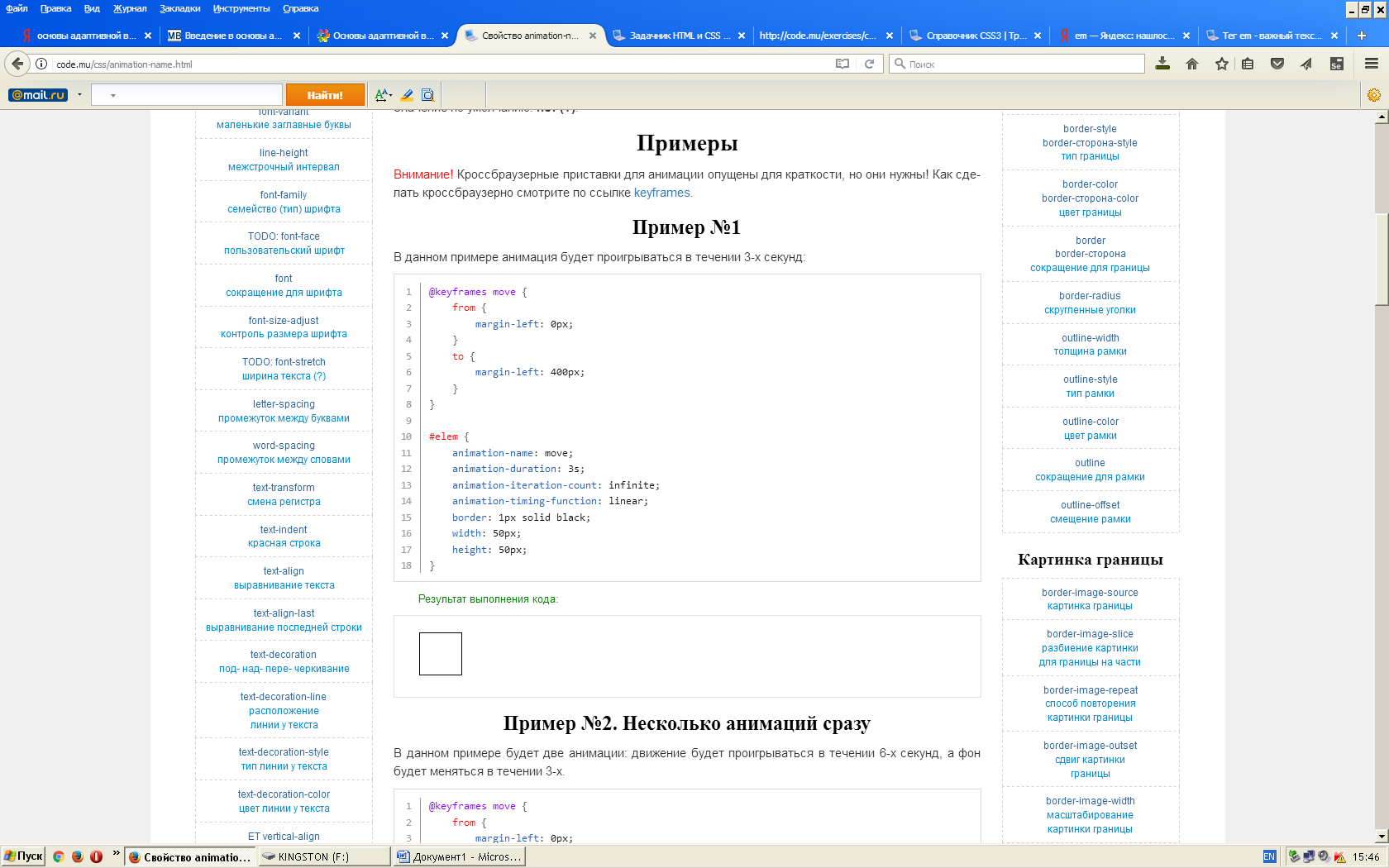
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок 2.animation-duration**

Свойство **animation-duration** устанавливает время выполнения анимации.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-duration: время в s или ms;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-duration:2s, 3s;}

**Значения**

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| s | Задает время в секундах (например 3s). |
| ms | Задает время в миллисекундах (например 3ms). Одна секунда - это 1000 миллисекунд. |

Значение по умолчанию: **0s**.

**Пример 1.**

В данном примере анимация будет проигрываться в течении 3-х секунд:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

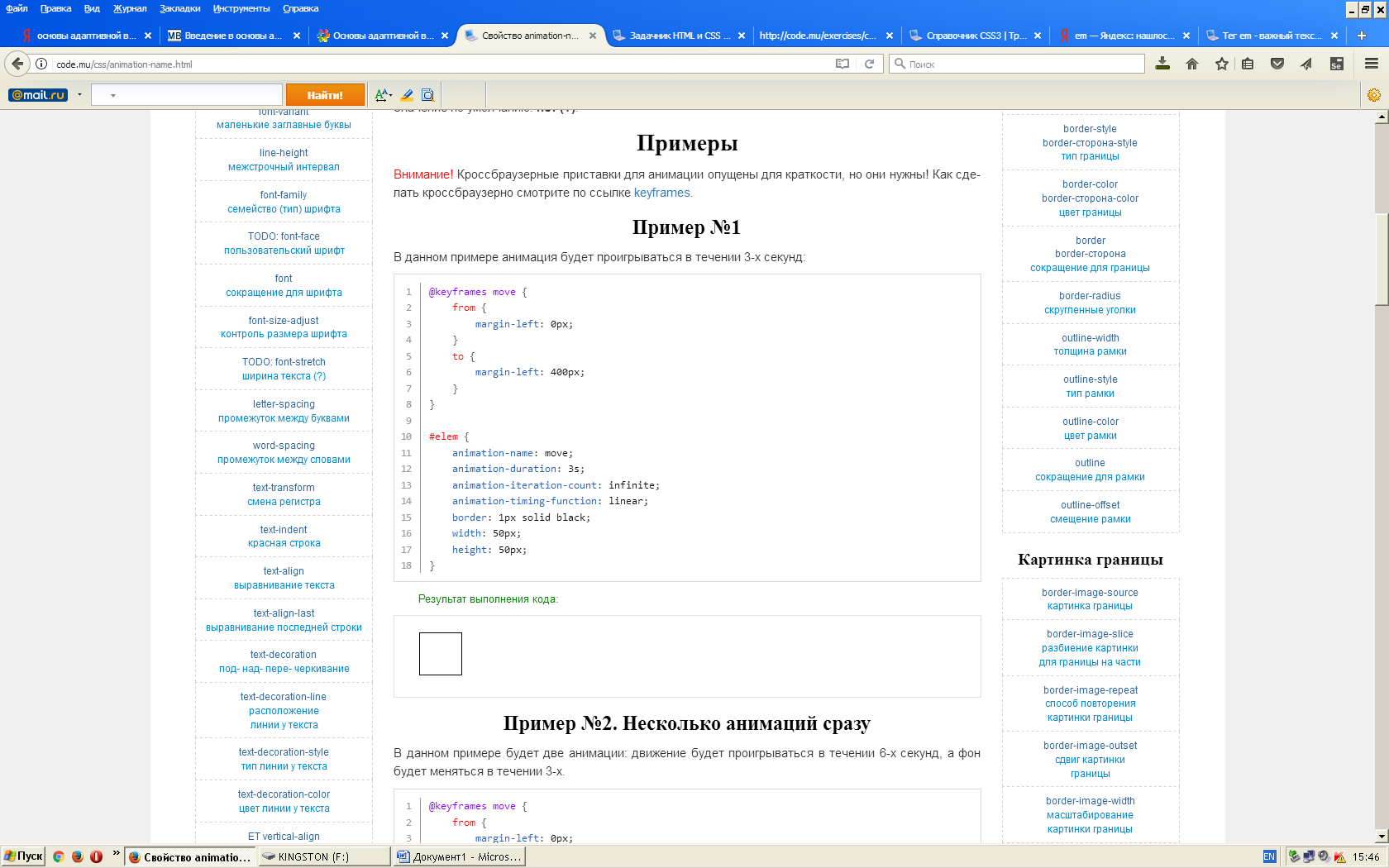
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 2. Несколько анимаций сразу**

В данном примере будет две анимации: движение будет проигрываться в течении 6-х секунд, а фон будет меняться в течении 3-х.

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

@keyframesbg{

from{

background-color:red;

}

to{

background-color:white;

}

}

#elem{

animation-duration:6s, 3s;

animation-name: move, bg;

animation-iteration-count: infinite, infinite;

animation-timing-function: linear, ease;

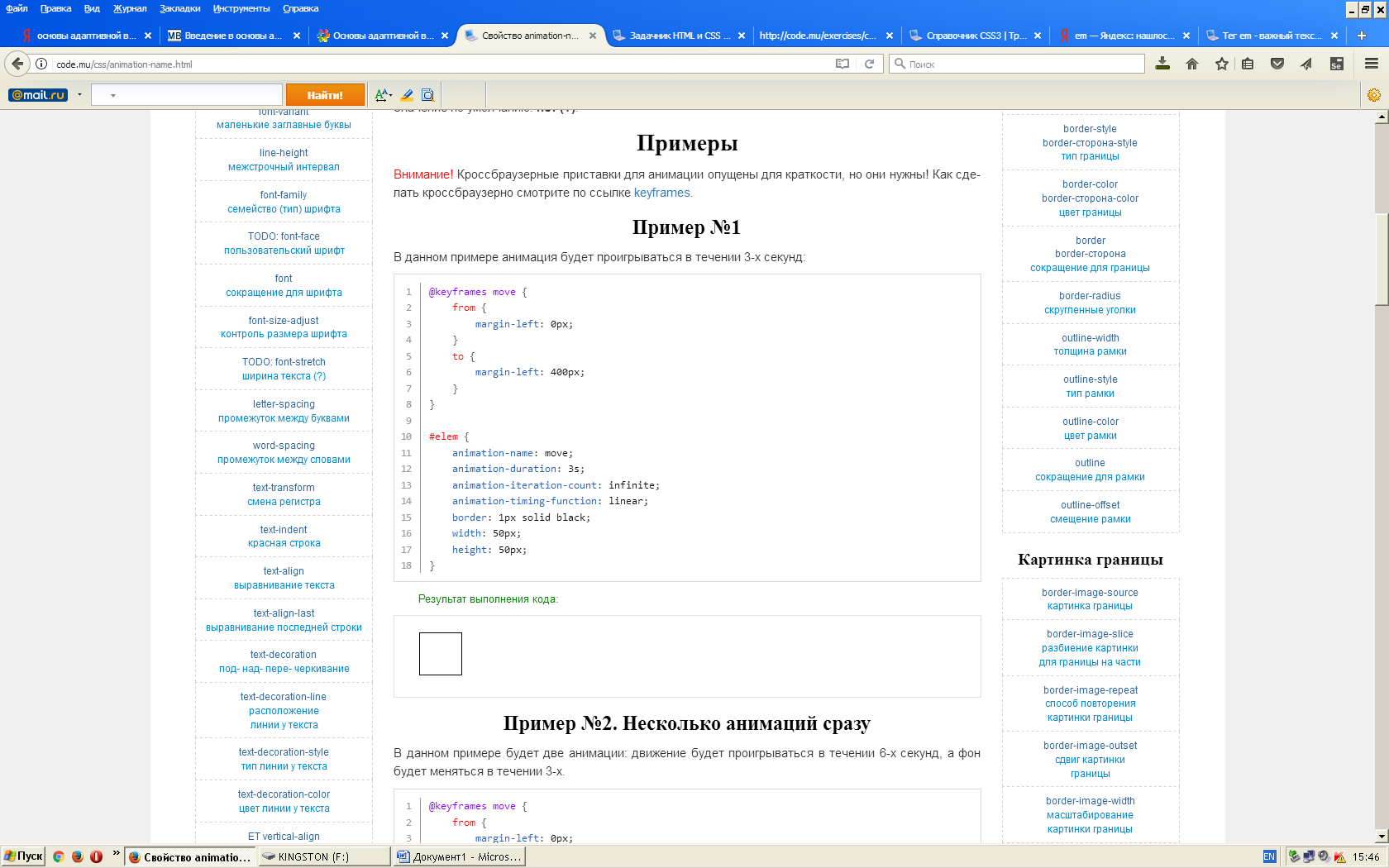
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок 3.animation-delay**

Свойство **animation-delay** устанавливает задержку перед запуском анимации.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-delay: время в s или ms;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-duration:2s, 3s;

}

**Значения**

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| s | Задает время в секундах (например 3s). |
| ms | Задает время в миллисекундах (например 3ms). Одна секунда - это 1000 миллисекунд. |

Значение по умолчанию: **0s**.

**Пример**

В данном примере перед анимацией будет задержка в 3 секунды (перед каждым новым проигрыванием), а затем анимация будет проигрываться в течении 6-х секунд:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-delay:3s;

animation-duration:6s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

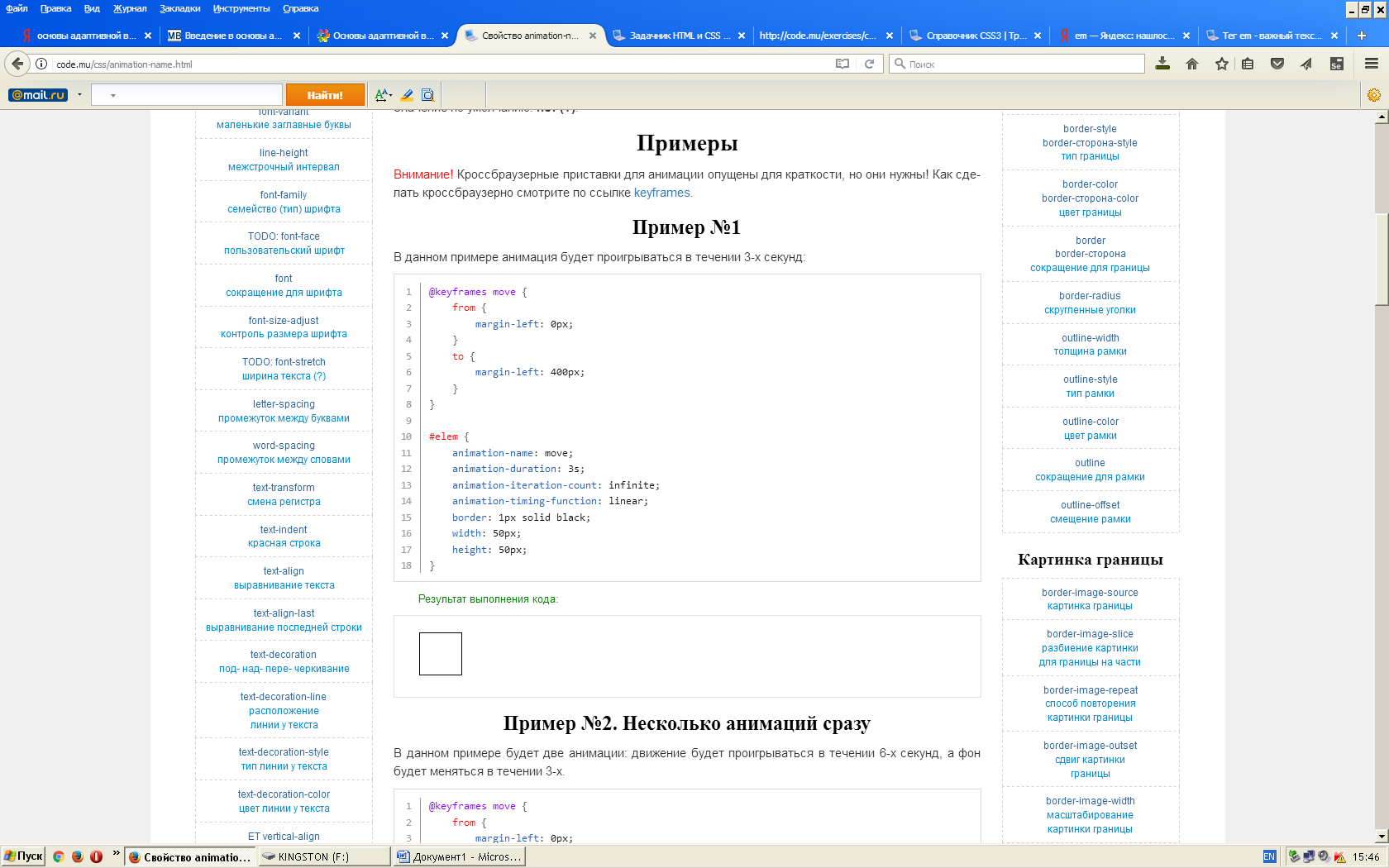
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок4.animation-timing-function**

Свойство **animation-timing-function** устанавливает скорость изменения (ускорение) анимации. Например, сначала медленно, потом быстро, потом медленно и т.д.

Данное свойство входит в свойство-сокращение animation.

## Синтаксис

Одноиззначений: ease, ease-in, ease-out, ease-in-out, linear, step-start, step-end, steps, cubic-bezier(параметры):

селектор{

animation-timing-function:значение;

}

Анимацияскачками:

селектор{

animation-timing-function:steps(числоскачков, start | end);

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-timing-function: ease, ease-in;;

}

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| ease | Сначала медленно, потом быстро, в конце опять медленно. |
| ease-in | Начинается медленно и постепенно ускоряется. |
| ease-out | Начинается быстро и постепенно останавливается. |
| ease-in-out | Сначала медленно, потом быстро, в конце опять медленно. От ease отличается скоростью. |
| linear | Всегда одна и та же скорость. |
| step-start | Анимации нет, свойство сразу принимает окончательное значение. |
| step-end | Анимации нет, свойство ждет время, заданное в animation-duraton, а затем мгновенно принимает окончательное значение. |
| steps | Значение свойства изменяется скачками. Применяется так: **animation-timing-function: steps(число скачков, start или end)** (см. примеры, чтобы понять разницу). |
| cubic-bezier | Кривая Безье, которая может задавать произвольную анимацию. См. генератор кривых Безье. |

Значение по умолчанию: **ease**.

**Сравнение различных значений**

ease

ease-in

ease-out

ease-in-out

linear

step-start

step-end

steps(3, end)

steps(3, start)

**Блок 5.animation-iteration-count**

Свойство **animation-iteration-count** задает количество повторов анимации. По умолчанию анимации повторится только 1 раз и затем вернется в исходное состояние. Однако,animation-iteration-count позволяет задать определенное число повторений или вообще задать бесконечное повторение (значение **infinite**). Интересно, что можно задавать **дробные значения** - в этом случае анимация проиграется лишь частично.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-iteration-count: целое или дробное число | infinite;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-duration: 3, 2, infinite;

}

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| Число | Заданное количество повторений, после совершения которых анимация остановится. Положение, которое она займет, регулируется свойством animation-fill-mode.  Можно задавать **дробные значения** - в этом случае анимация проиграется лишь частично. |
| infinite | Анимация будет повторятся бесконечно. |

Значение по умолчанию: **1**.

**Пример . Конечное число повторений**

В данном примере анимация повторится только 3 раза, а затем вернется в исходное состояние:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-duration:3s;

animation-name: move;

animation-iteration-count: 3;

animation-timing-function:linear;

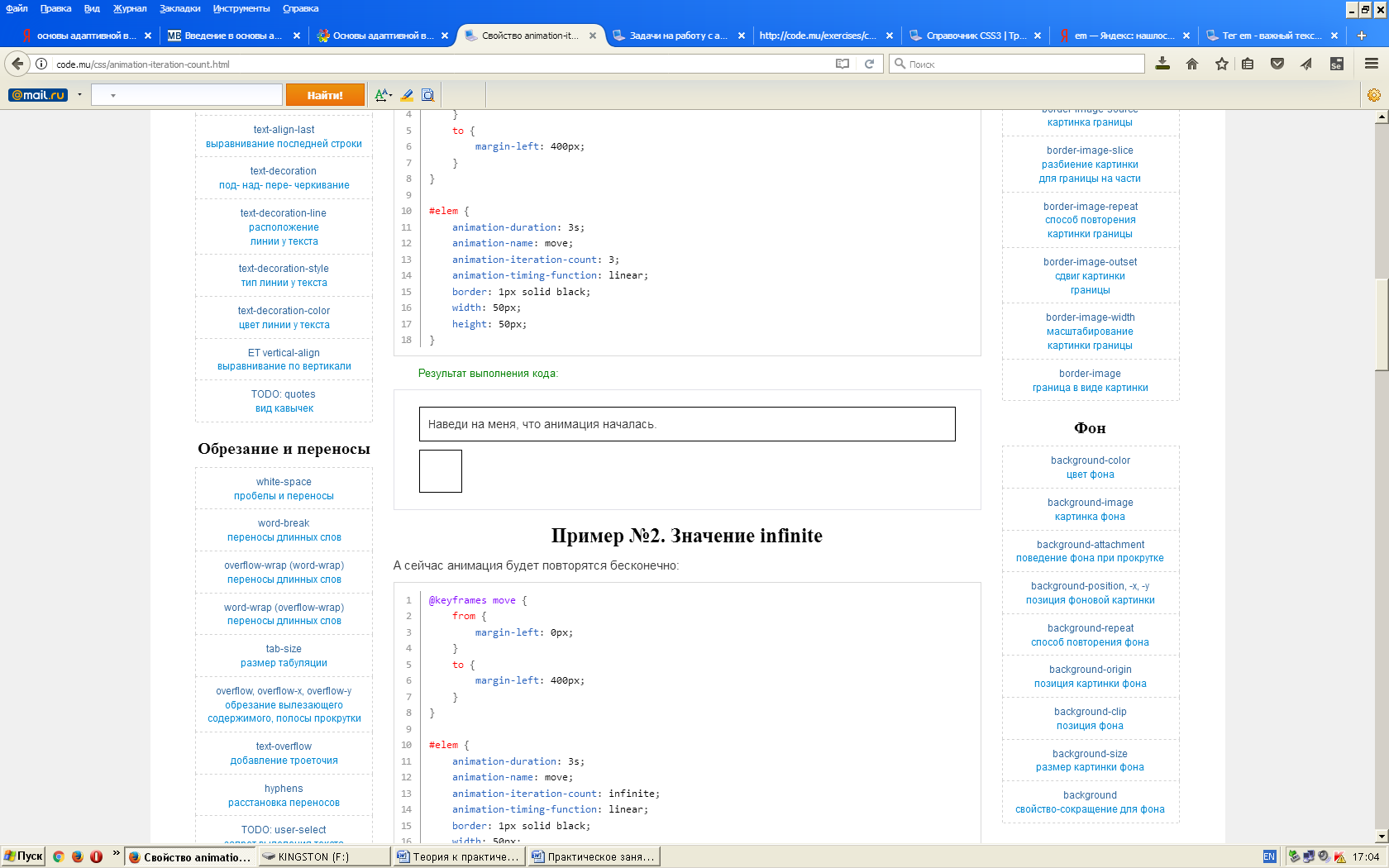
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:

****

**Пример . Значение infinite**

А сейчас анимация будет повторятся бесконечно:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

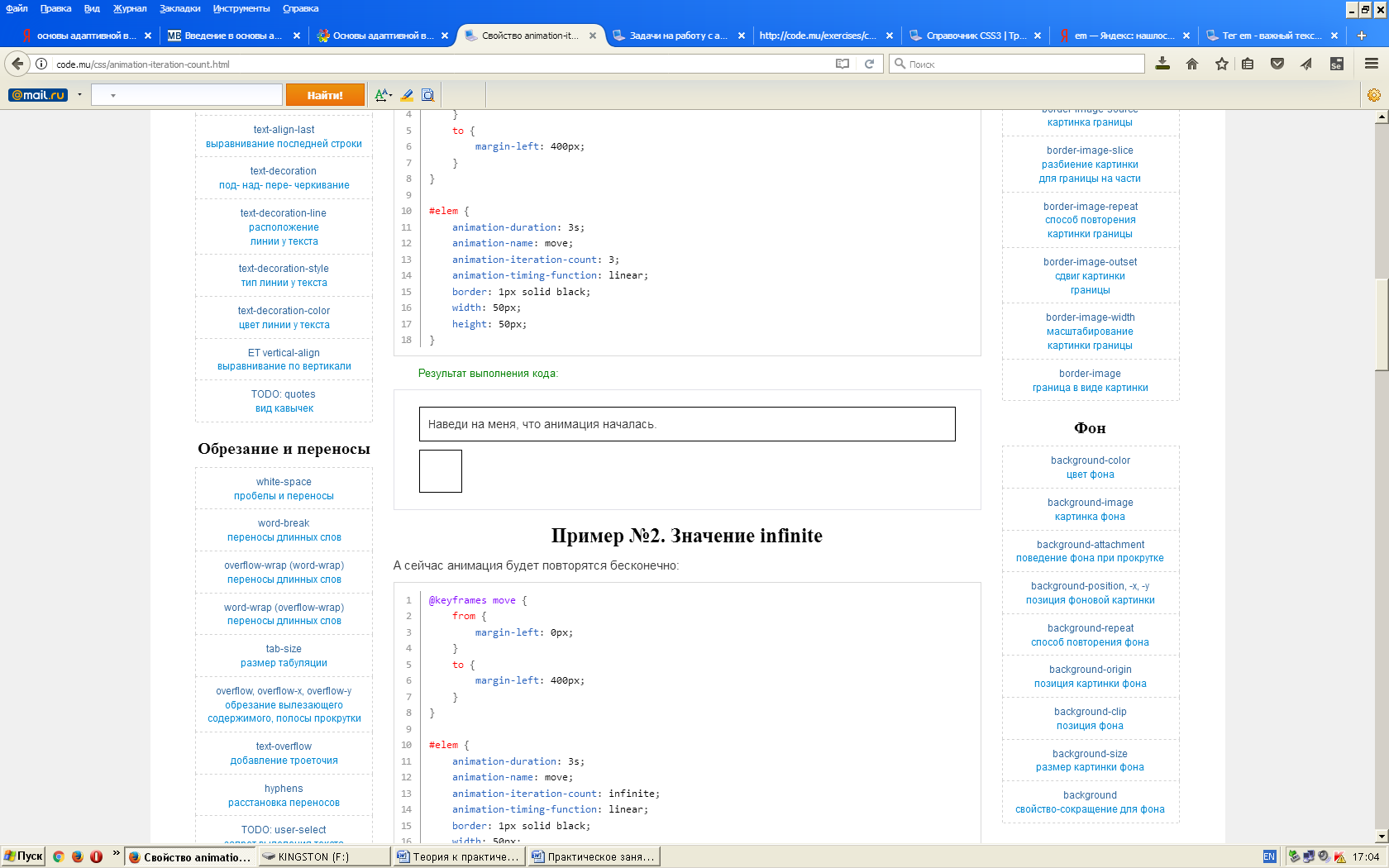
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок6. animation-direction**

Свойство**animation-direction**задаетнаправлениеанимации. По умолчанию анимация повторится только 1 раз и **затем вернется в исходное состояние**, затем, если задана задержка animation-delay, будет ждать заданное время и потом цикл начнется сначала. Данное свойство позволяет поменять это поведение, например, так чтобы анимация после окончания **оставалась в том месте, где она закончилась**, а не перескакивала в начальное положение. Также можно задать такое поведение: анимация дойдет до крайней точки и вернется обратно (как в transition). Смотрите описание ниже.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-direction: normal | reverse | alternate | alternate-reverse | initial | inherit;

}

Для анимирования нескольких свойств следует задавать значения через запятую:

селектор{

animation-duration: normal, reverse, alternate;

}

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| reverse | Анимация будет проигрываться в обратном направлении. |
| alternate | Анимация будет проигрываться сначала в прямом, а потом в обратном направлении (как transition). |
| alternate-reverse | Анимация будет проигрываться из конечного положения в начальное и обратно. По сути это значения **alternate** и **reverse** в одном флаконе. |
| initial | Устанавливает значение свойства в значение по умолчанию. |
| normal | Анимация будет проигрываться от начала к концу, а после окончания скачком перескакивать в начальное положение. |
| inherit | Наследует значение родителя. |

Значение по умолчанию: **normal**.

**Пример 1.**

Сейчас элемент будет двигаться в одну сторону, а потом возвращаться обратно, так как задано значение alternate. При этом animation-duration имеет значение 3 секунды и это значит, что перемещения "туда" и "обратно" будут длиться по 3 секунды.

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-direction: alternate;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

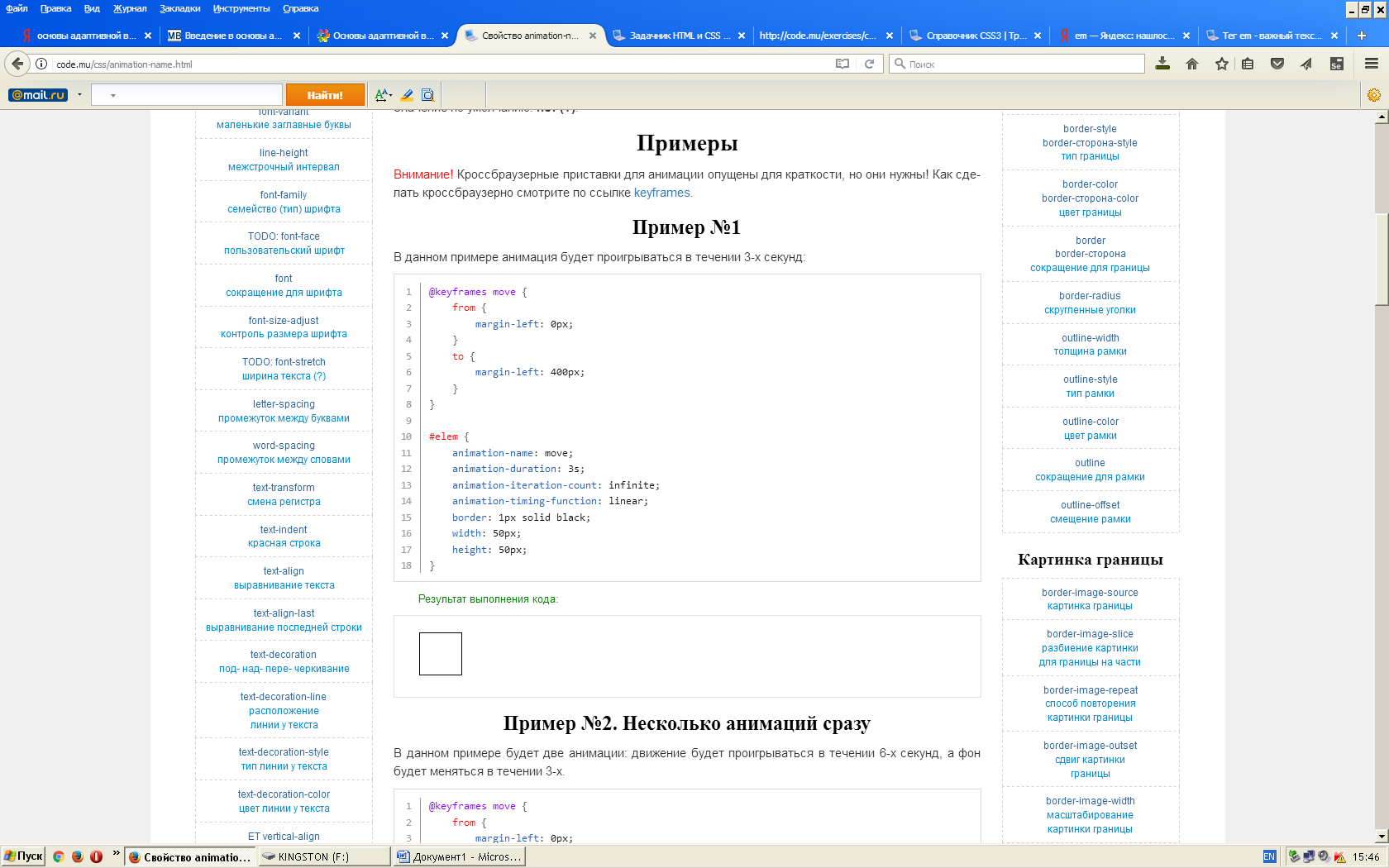
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 2.**

Сейчас элемент будет двигаться в **обратную сторону** (должен слева направо, а будет справа налево):

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-direction: reverse;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

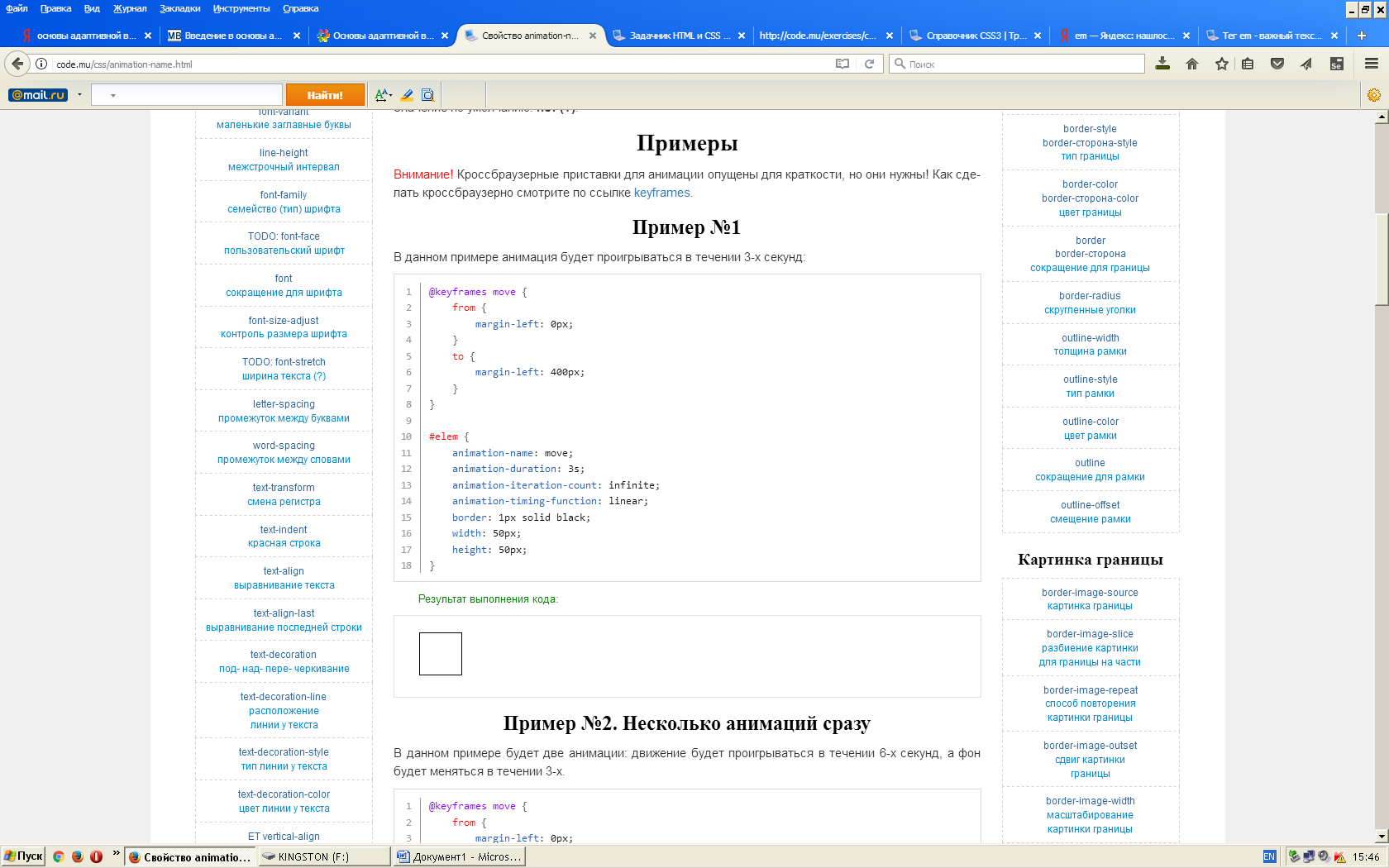
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 3.**

Сейчас элемент будет двигаться туда-сюда, но в обратном направлении (должен начинать слева, а будет начинать **справа**):

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

animation-direction: alternate-reverse;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

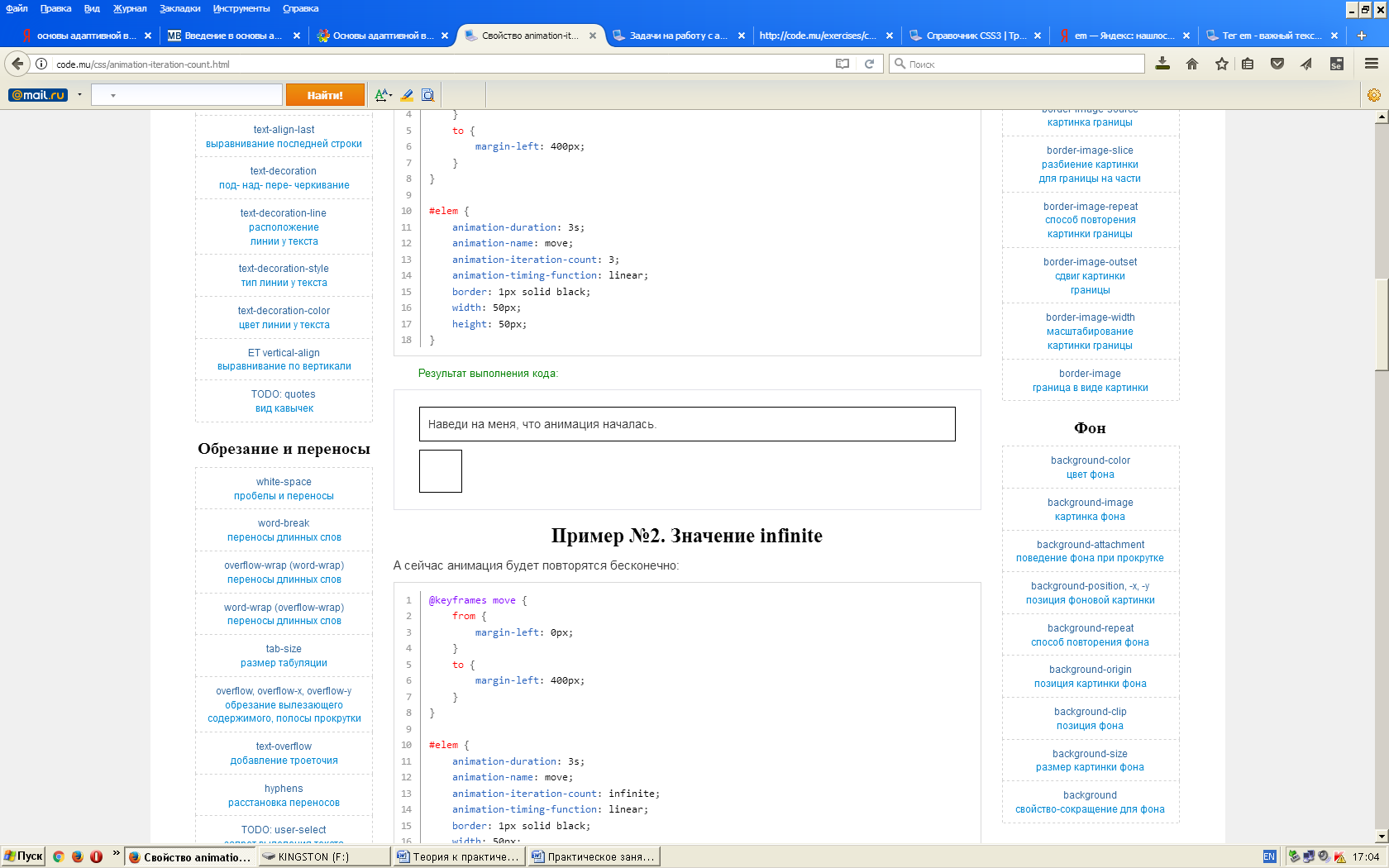
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок 7.animation-fill-mode**

Свойство **animation-fill-mode**устанавливает в каком положении останавливаться анимации после окончания. По умолчанию анимация возвращается в первоначальное значение и это может выглядеть не очень красиво.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-fill-mode: backwards | forwards | both | none;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор{

animation-duration: backwards, forwards, forwards;

}

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| none | Если установлена задержка анимации - то в течении времени задержки элемент будет оставаться на месте, а потом скачком перейдет к 0% кадру. После окончания анимации элемент не останется в том состоянии, где остановился, а перепрыгнет в начальное состояние. |
| backwards | Заставляет элемент двигаться после загрузки страницы к 0% кадру, даже если установлена задержка animation-delay, и оставаться там, пока не начнется анимация. После окончания анимации элемент не останется в том состоянии, где остановился, а перепрыгнет в начальное состояние. |
| forwards | Указывает браузеру, что после окончания анимации элемент останется в том состоянии, где остановился. |
| both | Включает в себя backwards и forwards - после загрузки страницы элемент установится к 0% кадру, а после окончания анимации элемент останется там, где остановился. |

Значение по умолчанию: **none**.

**Пример 1. Значение none**

В данном примере margin-leftустановлен в значение **300px** для элемента, а для первого кадра анимации - в 0px. Так как нет задержки animation-delay, то элемент после загрузки страницы станет в **0px**, а не в **300px**. Кроме того,анимация после окончания будет перепрыгивать в начальное значение:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: none;

animation-delay:0s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

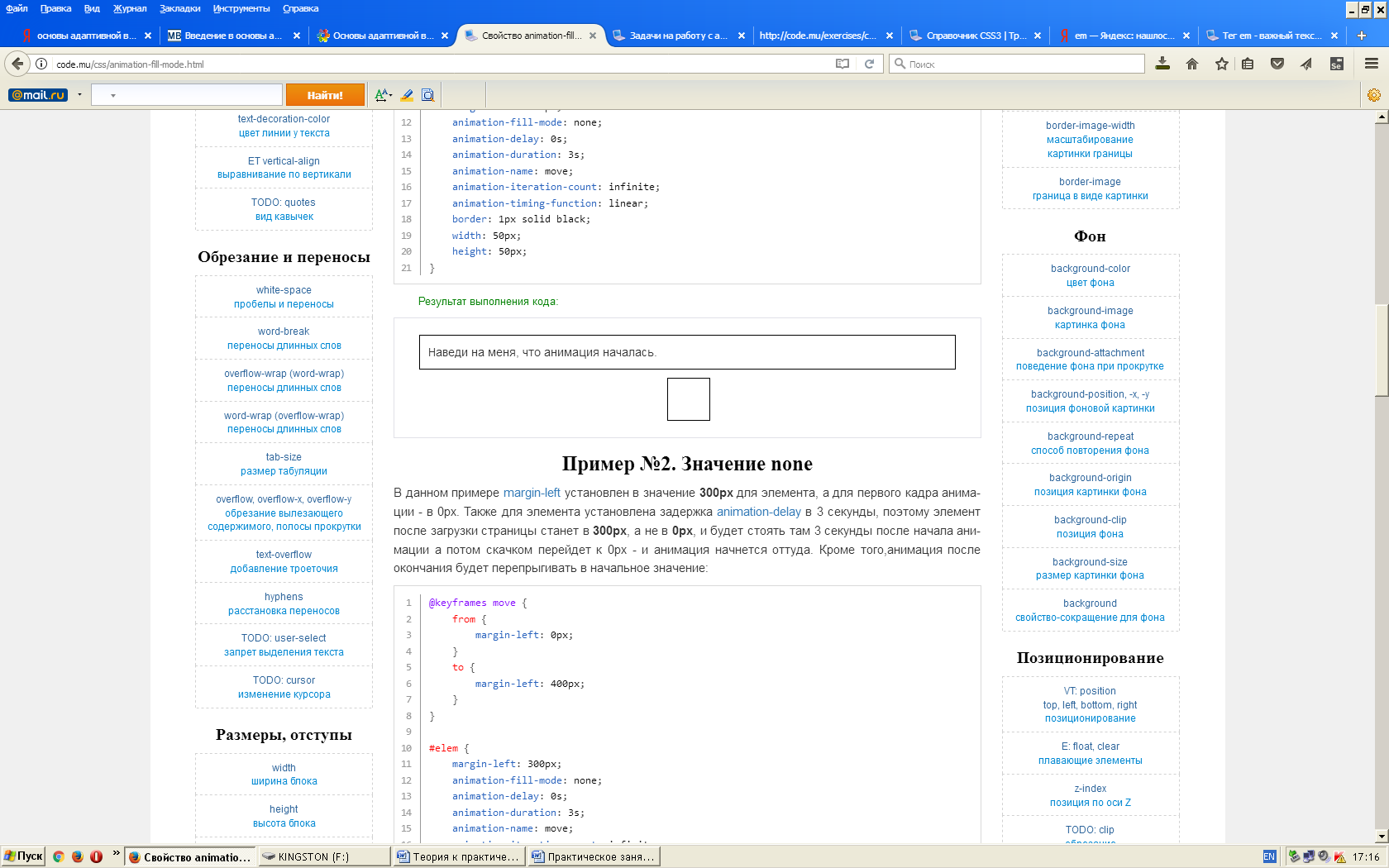
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:

****

**Пример 2. Значение none**

В данном примере margin-leftустановлен в значение **300px** для элемента, а для первого кадра анимации - в 0px. Также для элемента установлена задержка animation-delay в 3 секунды, поэтому элемент после загрузки страницы станет в **300px**, а не в **0px**, и будет стоять там 3 секунды после начала анимации а потом скачком перейдет к 0px - и анимация начнется оттуда. Кроме того,анимация после окончания будет перепрыгивать в начальное значение:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: none;

animation-delay:0s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

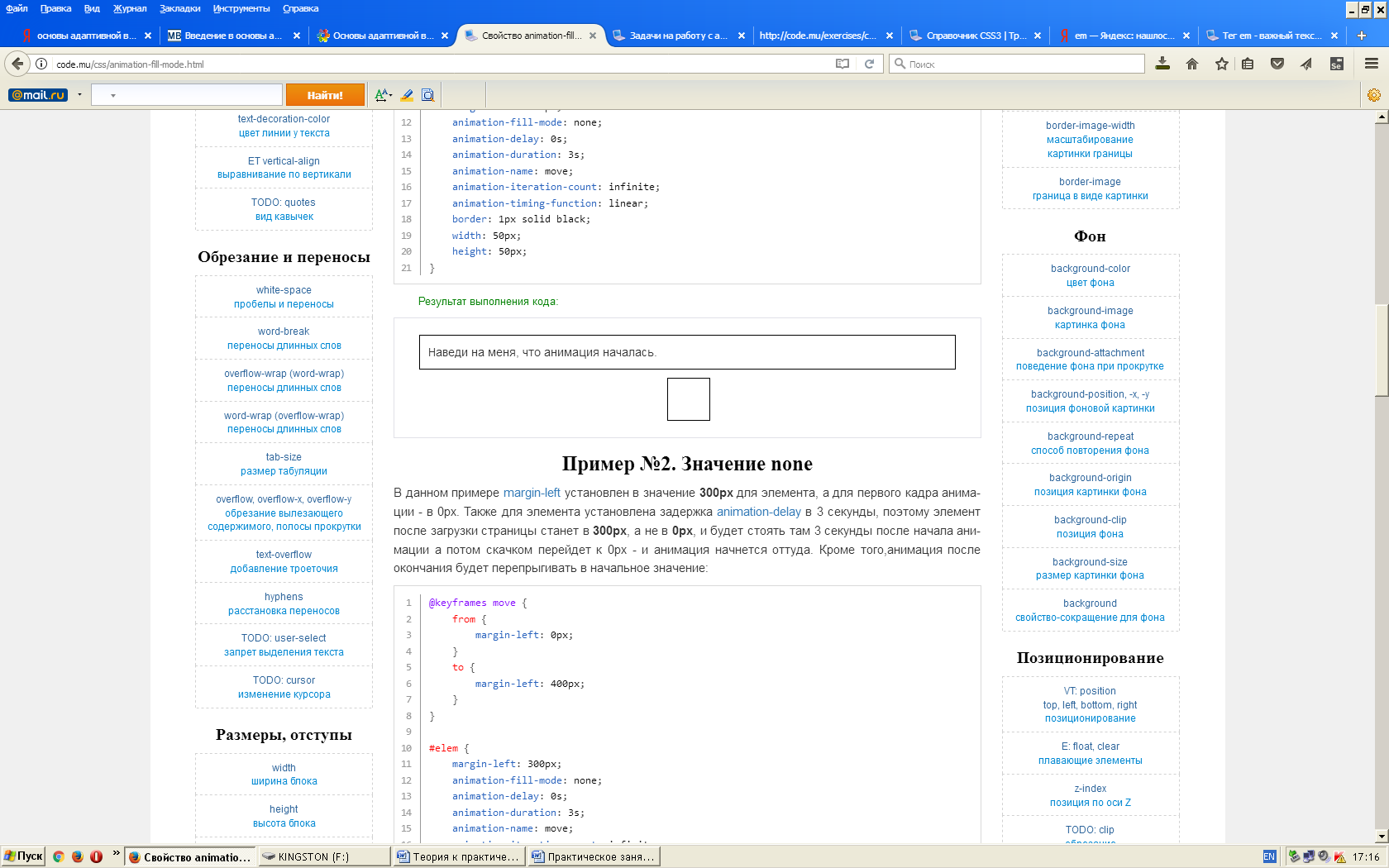
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 3. Значение backwards**

В данном примере margin-leftустановлен в значение 300px для элемента, а для первого кадра анимации - в **0px**. Также для элемента установлен **animation-fill-mode** в значении **backwards**, поэтому элемент после загрузки страницы станет в **0px**, а не в **300px**, как это было для **animation-fill-mode** в значении none. Кроме того,анимация после окончания будет перепрыгивать в начальное значение:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: backwards;

animation-delay:0s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

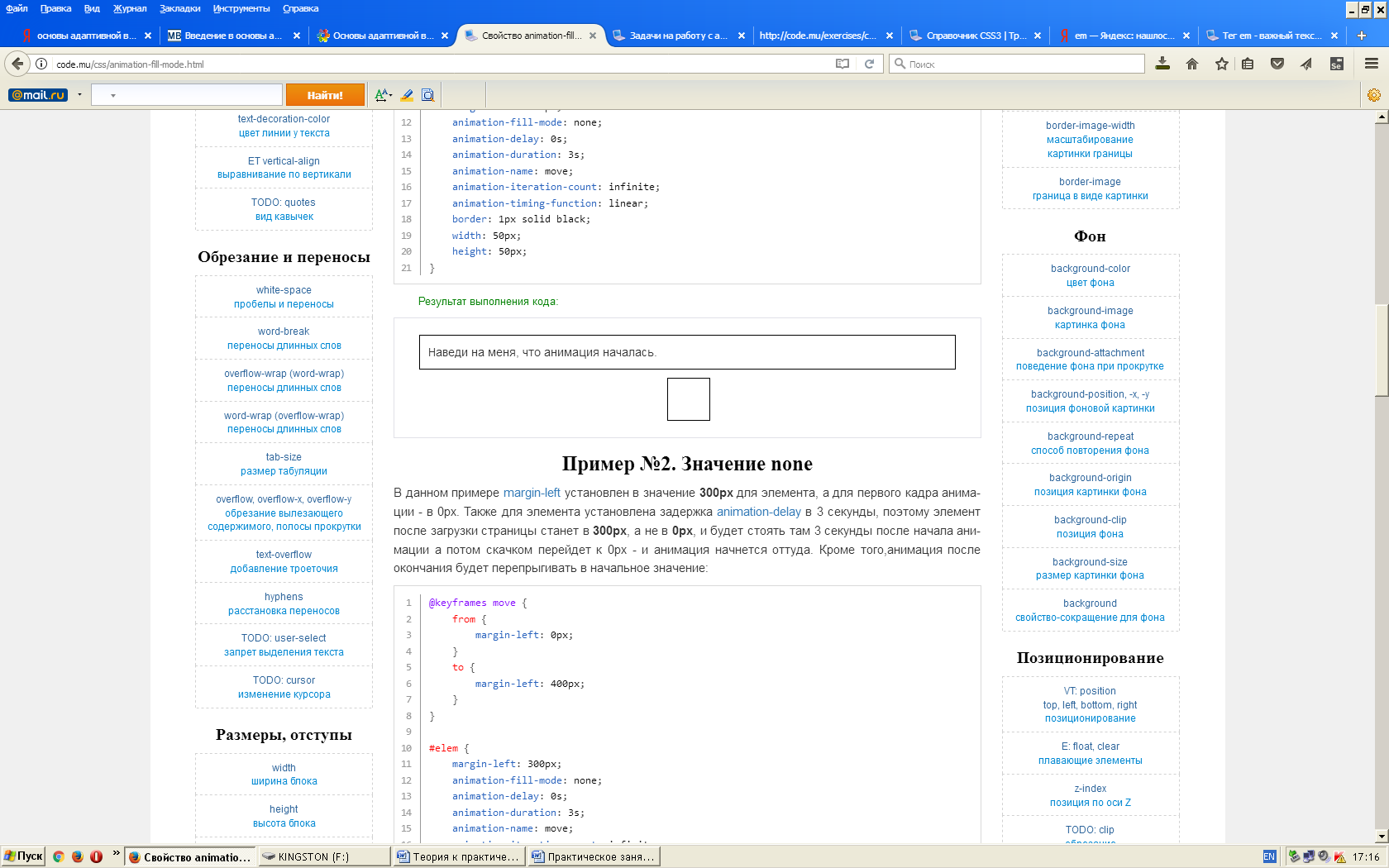
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 4. Значение none и количество повторений равно 1**

В данном примере margin-leftустановлен в значение **300px** для элемента, а для первого кадра анимации - в **0px**. Также для элемента установлена задержка [animation-delay](http://code.mu/css/animation-delay.html) в 3 секунды, поэтому элемент после загрузки страницы станет в **300px**, а не в **0px**, и будет стоять там 3 секунды после начала анимации а потом скачком перейдет к 0px - и анимация начнется оттуда. Так как **animation-fill-mode** установлен в значении none, то после проигрывания анимации элемент вернется в начальное значение:

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: none;

animation-delay:3s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: 1;

animation-timing-function:linear;

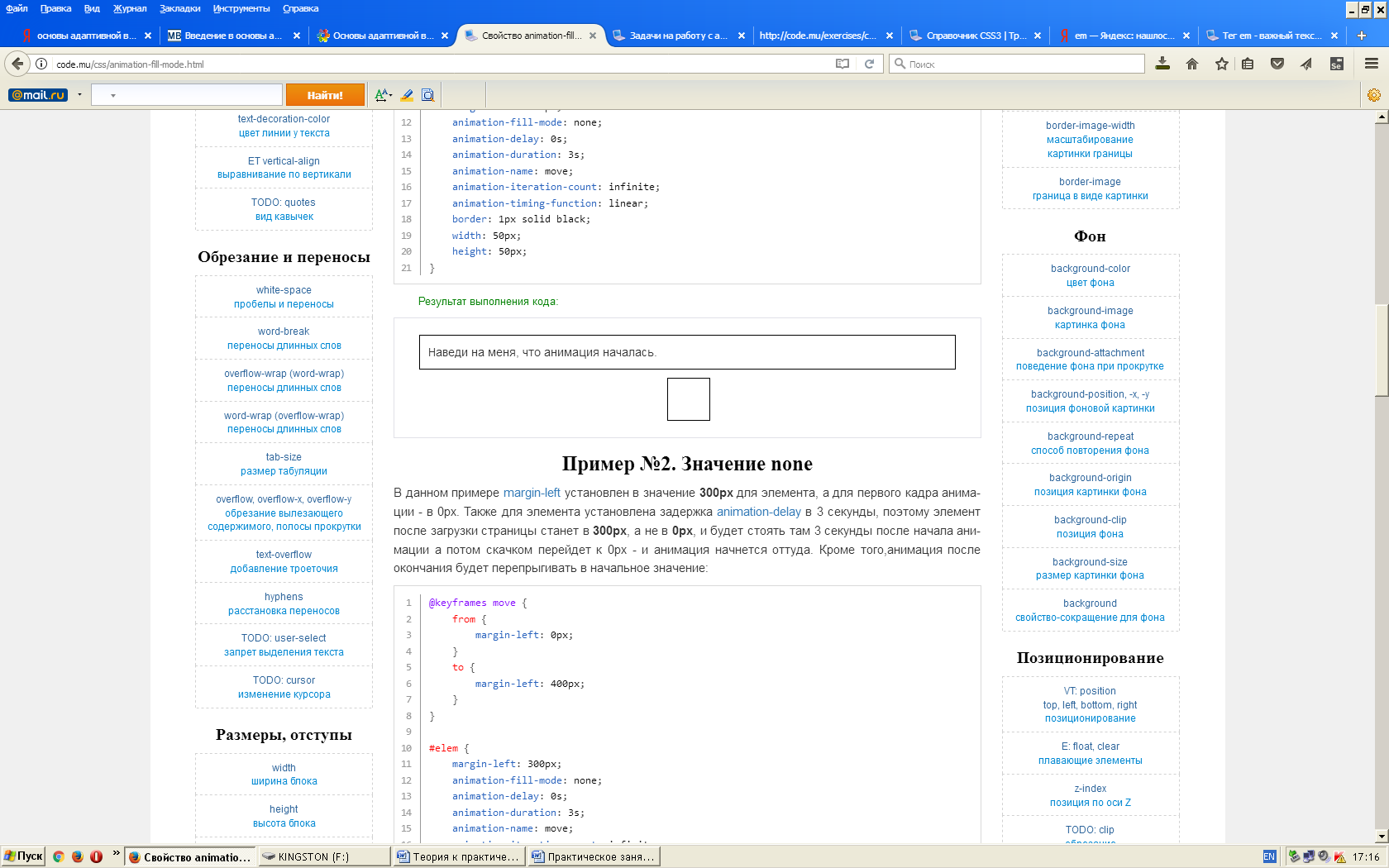
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример 5. Значение forwards и количество повторений равно 1**

В данном примере **animation-fill-mode**установлен в значении **forwards**, а число повторений анимации - в 1. После проигрывания анимации элемент останется в том положении, где закончилась анимация, а не прыгнет в исходную точку.

Кроме того, **margin-left**установлен в значение **300px** для элемента, а для первого кадра анимации - в 0px. Также для элемента установлена задержка **animation-delay** в 3 секунды, поэтому элемент после загрузки страницы станет в 300px, а не в 0px, и будет стоять там 3 секунды после начала анимации а потом скачком перейдет к 0px - и анимация начнется оттуда.

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: forwards;

animation-delay:3s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: 1;

animation-timing-function:linear;

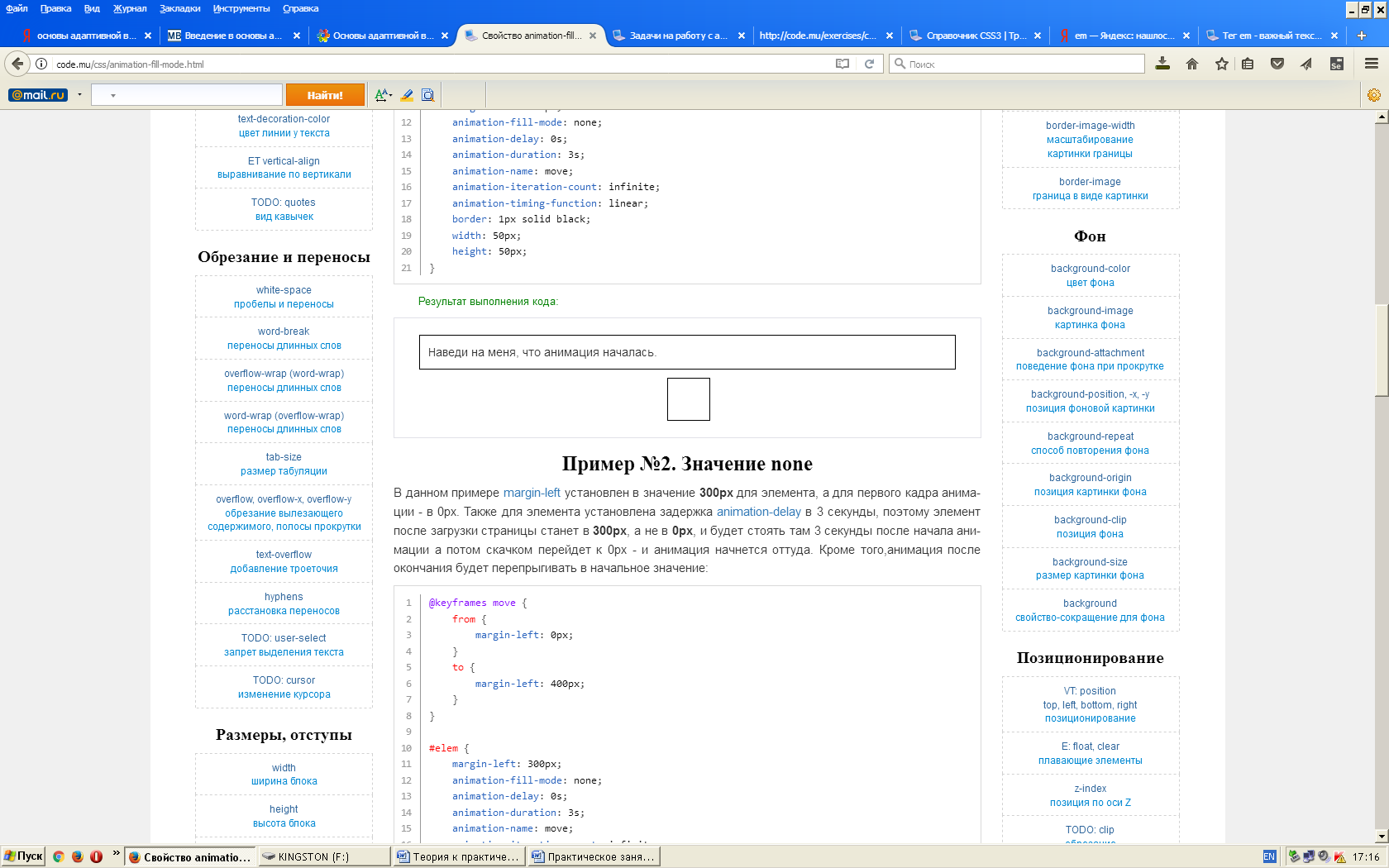
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:

****

**Пример 6. Значение both и количество повторений равно 1**

В данном примере **animation-fill-mode**установлен в значении **both**, а число повторений анимации- в 1. После загрузки страницы элемент будет стоять в первом кадре анимации (в 0px, а не в 300px), а после проигрывания анимации элемент останется в том положении, где закончилась анимация, а не прыгнет в исходную точку.

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#elem{

margin-left: 300px;

animation-fill-mode: both;

animation-delay:3s;

animation-duration:3s;

animation-name: move;

animation-iteration-count: 1;

animation-timing-function:linear;

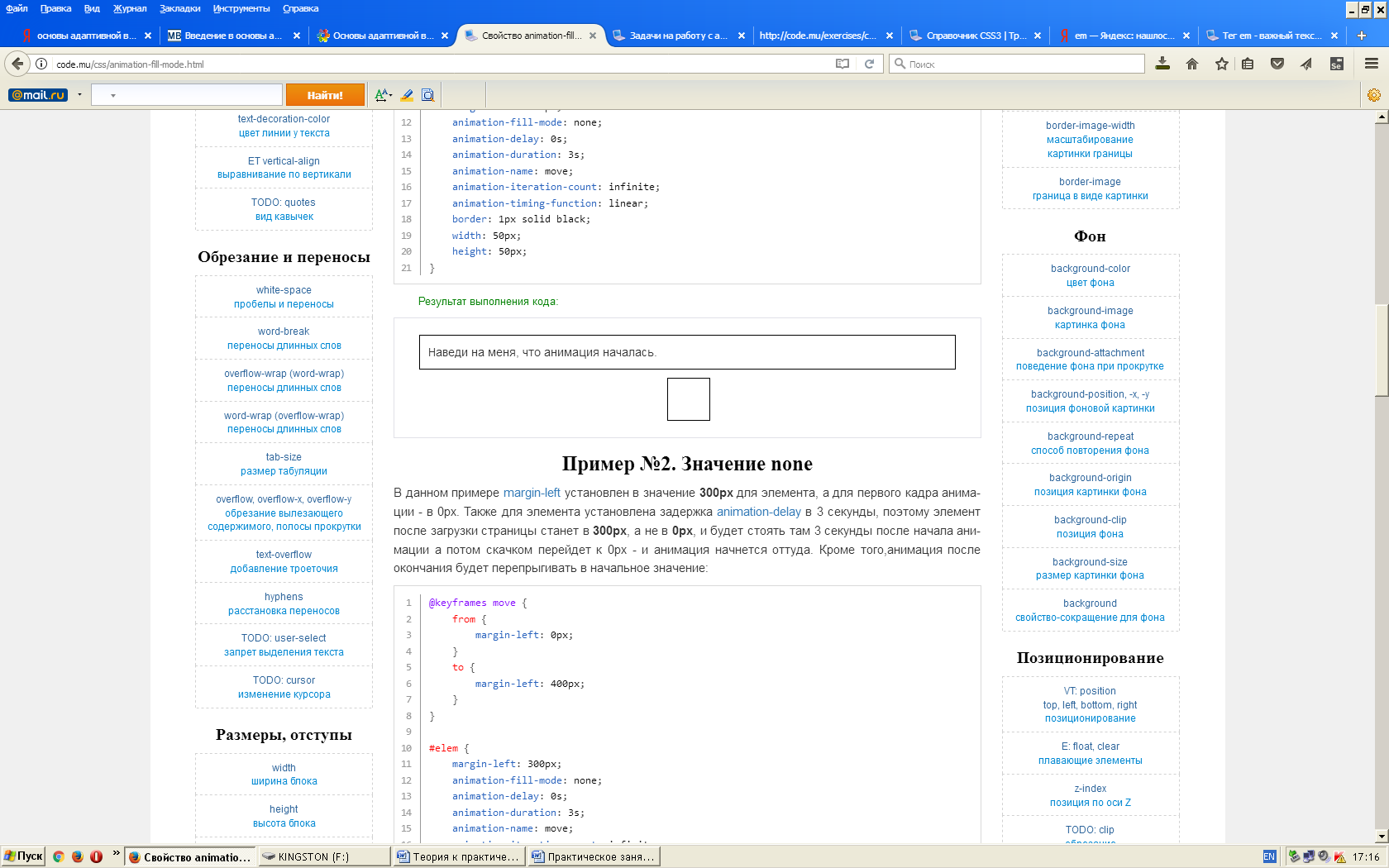
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Блок 8. animation-play-state**

Свойство **animation-play-state** позволяет задать состояние анимации: она проигрывается или стоит на паузе.

Данное свойство входит в свойство-сокращение animation.

**Синтаксис**

селектор{

animation-play-state: paused | running;

}

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| paused | Анимация стоит на паузе. |
| running | Анимация проигрывается. |

Значение по умолчанию: **running**.

**Пример 1.**

В данном примере анимация будет запускаться по наведению мышкой на элемент, а прекращаться, когда мышь будет убираться с элемента:

@keyframes move{

from{

width: 200px;

}

to{

width: 400px;

}

}

#anim{

animation-play-state: paused;

animation-duration:1s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

border: 1px solid black;

height: 50px;

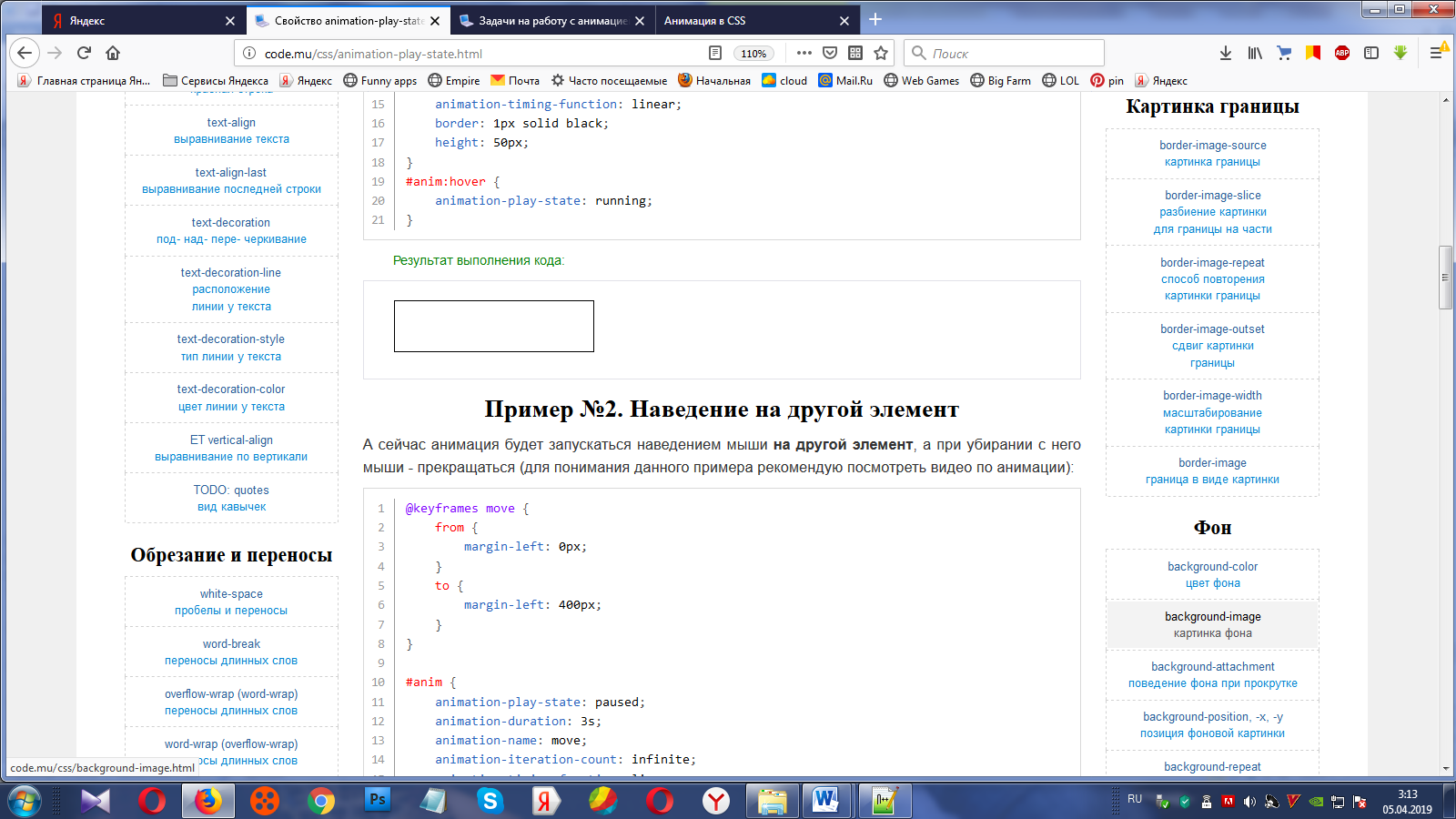
}

#anim:hover{

animation-play-state: running;

}

Результат выполнения кода:



**Пример 2.. Наведение на другой элемент**

А сейчас анимация будет запускаться наведением мыши **на другой элемент**, а при убирании с него мыши - прекращаться (для понимания данного примера рекомендую посмотреть видео по анимации):

@keyframes move{

from{

margin-left: 0px;

}

to{

margin-left: 400px;

}

}

#anim{

animation-play-state: paused;

animation-duration:3s;

animation-name: move;

animation-iteration-count: infinite;

animation-timing-function:linear;

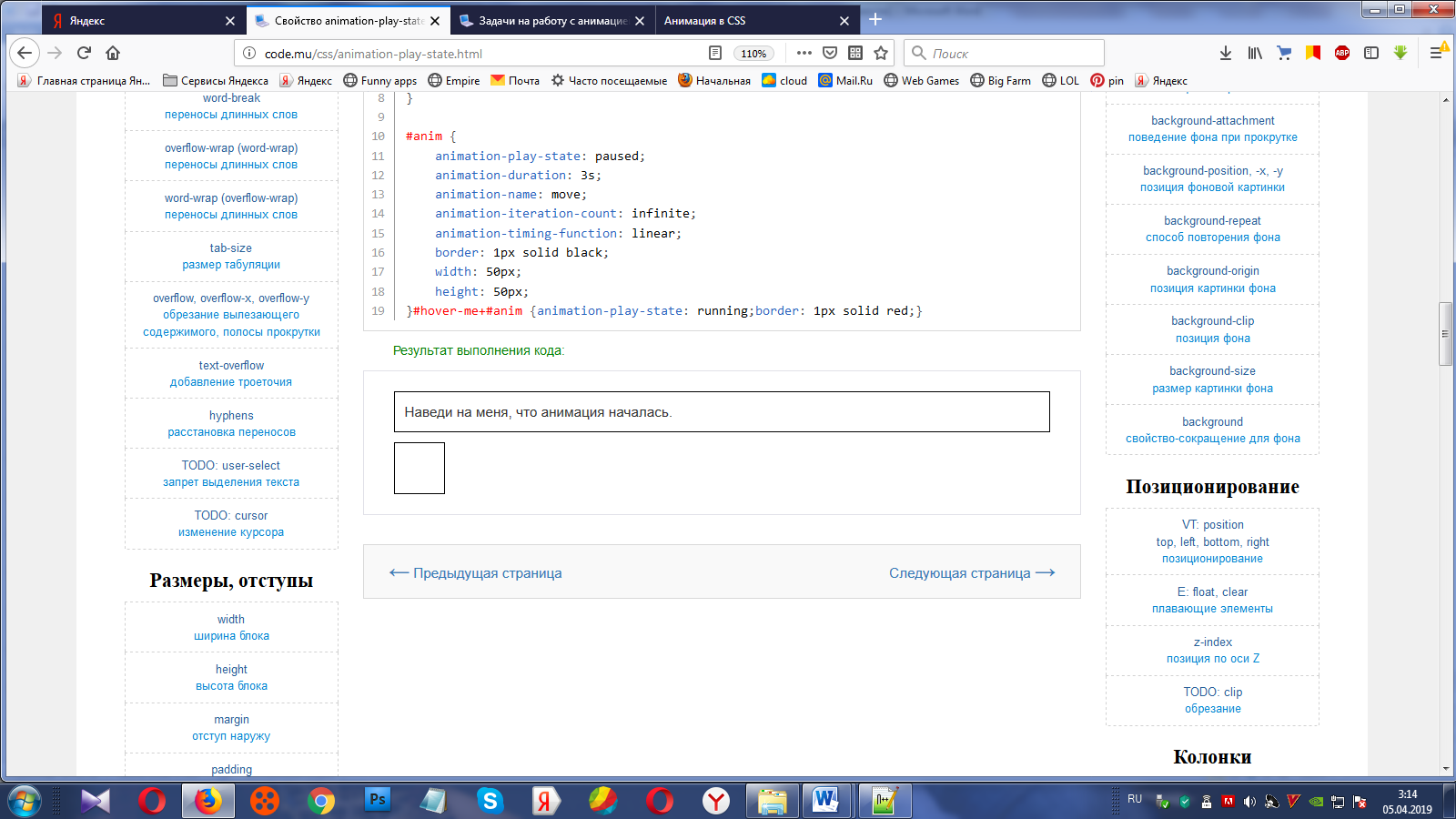
border: 1px solid black;

width: 50px;

height: 50px;

}#hover-me+#anim{animation-play-state:running;border: 1px solid red;}

Результат выполнения кода:



**Блок9.animation**

Свойство**animation** - этосвойствосокращениедляанимации, позволяетзадатьвсесвойствадляанимацииодновременно: animation-name, animation-duration, animation-delay, animation-timing-function, animation-iteration-count, animation-direction, animation-fill-mode, animation-play-state. Порядок значения не имеет, время выполнения анимации **animation-duration** обязательно должно стоять перед задержкой **animation-delay**. Обязательными являются только свойства **animation-name** и **animation-duration**.

**Синтаксис**

селектор {

animation: animation-name | animation-duration | animation-delay | animation-timing-function | animation-direction | animation-fill-mode | animation-play-state;

}

Для добавления нескольких анимаций следует задавать значения через запятую:

селектор {

animation: anim1 3s ease, anim2 5s ease-in 6s;

}

**Значения**

Значение **none** отключает анимацию совсем.

Значение по умолчанию: **none**.

**Пример 1.**

В данном примере перед анимацией будет задержка в 3 секунды (перед каждым новым проигрыванием), а затем анимация будет проигрываться в течении 6-х секунд:

@keyframes move {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

#elem {

animation: move 6s infinite 3s linear;

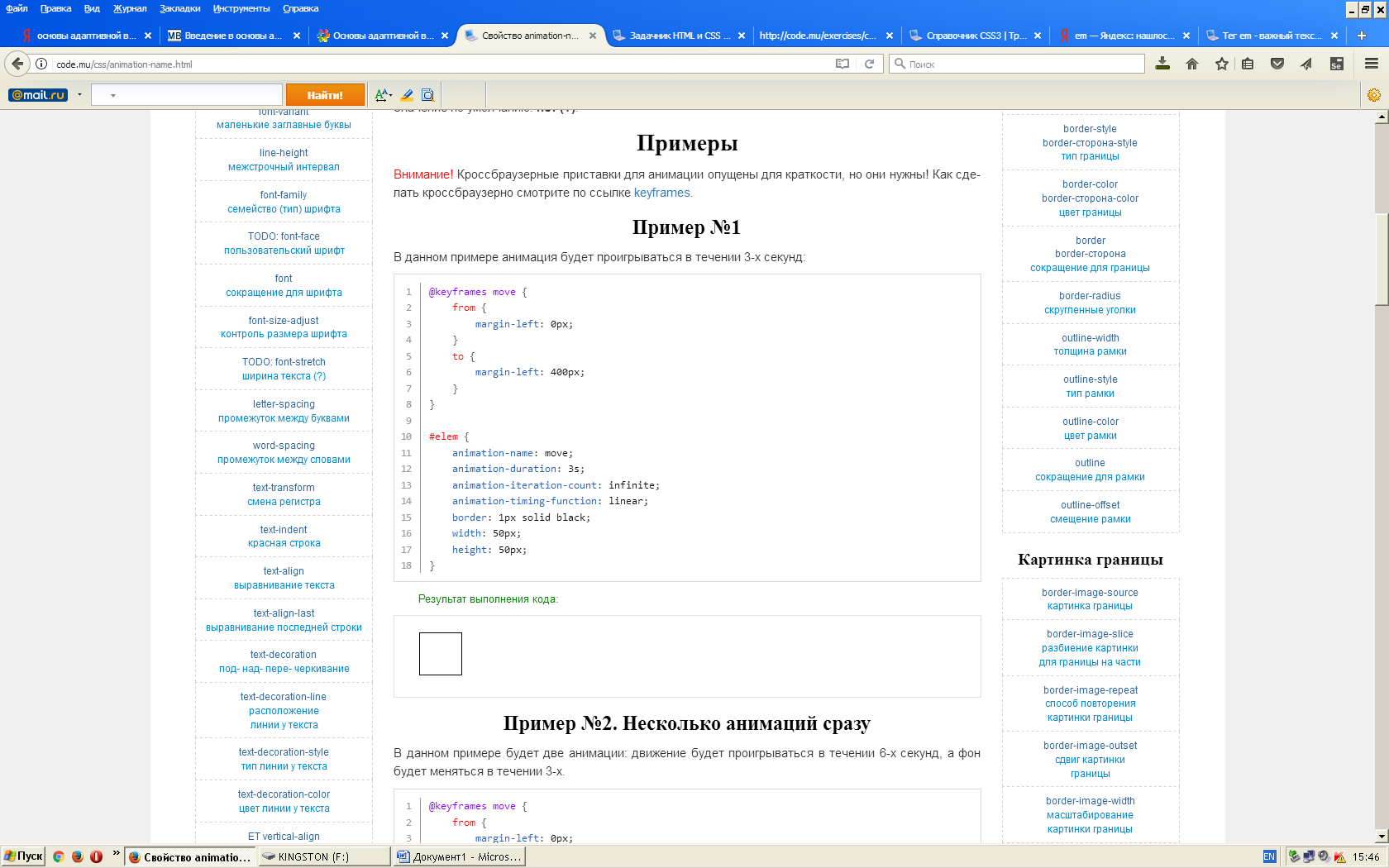
border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:



**Пример . Возможные проблемы**

Анимация, записанная в краткой форме, может не работать из-за своего названия. Давайте рассмотрим следующую анимацию:

селектор {

@keyframesreverse{

from{

left: 0;

}

to{

left: 300px;

}

}

.example{

animation: reverse 2s 1s;

}

}

Заметьте, я использую название анимации **reverse**. На первый взгляд, все хорошо, но aнимация не работает, т.к. **reverse** — валидное ключевое слово для свойства animation-direction. Так же анимация не будет работать при использовании в краткой форме в названии других ключевых слов. Но все работает хорошо при использовании 'полной' формы описания.

К ключевым словам-значениям animation-direction, ломающими анимации, стоит отнести и ключевые слова, относящиеся к функциям сглаживания, а так же **infinite**, **alternate**, **running**, **paused** и так далее.

**Блок10.@keyframes**

Команда **@keyframes** задает ключевые кадры анимации.

**Синтаксис**

селектор {

@keyframesимя\_анимации{ключевые кадры};

}

**Пример**

Задаем ключевые кадры:

@-o-keyframesanim {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

@-ms-keyframesanim {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

@-moz-keyframesanim {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

@-webkit-keyframesanim {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

@keyframesanim {

from {

margin-left: 0px;

}

to {

margin-left: 400px;

}

}

#elem {

-o-animation: anim 3s infinite;

-ms-animation: anim 3s infinite;

-moz-animation: anim 3s infinite;

-webkit-animation: anim 3s infinite;

animation: anim 3s infinite;

border: 1px solid black;

width: 50px;

height: 50px;

}

Результат выполнения кода:

