

Горячее

Лучшее

Свежее

...



empenoso



MS, Libreoffice &amp; Google docs

## Как я научил Google Календарь показывать возраст именинников

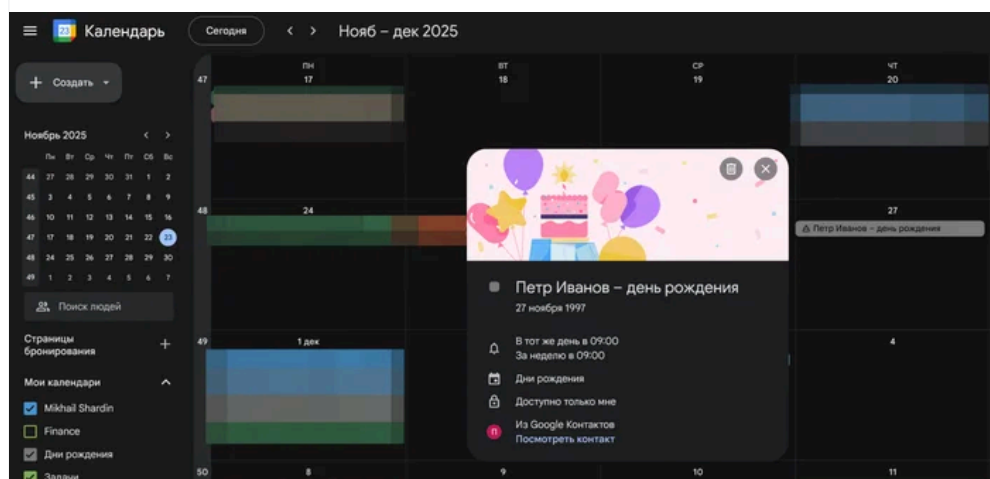


15 дней назад



3.7K

Если вы пользуетесь Гугл календарём, то стандартное напоминание выглядит как «ДР у Петра» и очень хорошо что Гугл теперь отображает и саму дату рождения — ещё несколько лет назад этого не было. Приходилось гадать — сколько лет-то человеку?



Стандартное отображение Гугл календаря в 2025 году о дне рождения

Хотя задача упрощается и дата рождения уже перед глазами, а контакт можно открыть одним кликом, но всё равно приходится считать в уме — это круглая дата или нет?

В 2025 году с отображением даты рождения стало гораздо проще, но проблема стара как сам Google Calendar. В 2019 году я уже писал о попытках решить её разными способами: через громоздкие скрипты и старые методы [Calendar API в 2022 году](#). Но многое из того давно сломалось, а Calendar API устарело.

Поэтому сейчас решил сделать через People API аккуратную автоматизацию, которая будет показывать в календаре не только «ДР у Ивана», но и сколько ему исполняется.



**Используйте аккаунт Яндекса для входа на сервис**  
Безопасный вход без дополнительной регистрации на сайте

Войти с Яндекс ID

Логин

Пароль

Войти

Создать аккаунт

Забыли пароль?

или продолжите с



Войти с Яндекс ID



Войти через VK ID

Промокоды

Работа

Курсы

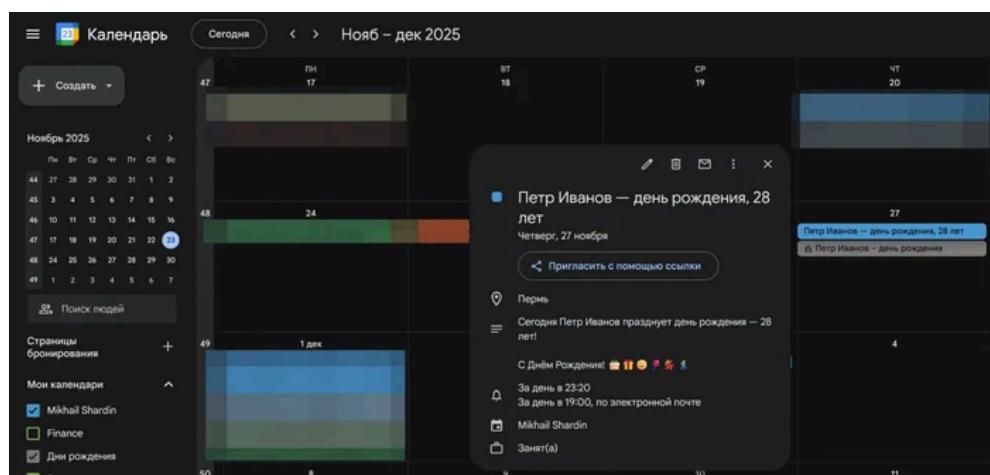
Реклама

Игры

Пополнение Steam







Результат работы скрипта

## Что мы получим в итоге

На скриншоте выше — результат работы скрипта: в календаре вы видите не просто «ДР у Петра», а строку вида «Петр — 28 лет». Никаких подсчётов в уме, никаких переходов в карточку контакта — нужная информация появляется прямо в событии.

Главные фишки новой версии:

- People API — работает в отличии от Calendar API.
- CONFIG-файл — меняете настройки без погружения в код. Даже если вы далеки от программирования, всё сводится к паре значений.
- Русский язык — корректные склонения «год/года/лет», без костылей.
- Телефон в описании — можно позвонить имениннику прямо из уведомления календаря, если номер есть в контактах. На скриншоте несуществующий для теста человек и телефона у него понятно нет.

И маленькая подготовка, чтобы всё завелось. Данные — это топливо: проверьте, что у контактов указан полный год рождения. Если года нет — скрипт не сможет вычислить возраст, и никакой магии не произойдёт.

## Инструкция: «Копировать — Вставить — Забыть»

Самая «сложная» часть — это не написание кода, а преодоление страха перед пустым редактором. Мы пойдем по пути наименьшего сопротивления.

1. Откройте [script.google.com](https://script.google.com) и нажмите большую кнопку «Создать проект».
2. Удалите всё, что есть в редакторе и вставьте код, приведенный ниже.

```
/**
```

```
* @fileoverview Скрипт для Google Apps Script, который создает в Google Календаре события
```

```
* о днях рождения контактов с указанием их возраста.
```

```
*
```

```
* @VERSION 2.0
```

```
* @Author Mikhail Shardin
```

**Пикабу Игры**  
+1000 бесплатных  
онлайн игр



**Рыбный дождь**

Спорт, Симуляторы, Рыбалка

Играть

Реклама

Топ прошлой недели

solenakrivetka  
7 постов

Animalrescued  
53 поста



```

* @see https://shardin.name/
*/

// --- НАСТРОЙКИ СКРИПТА ---
const CONFIG = {
// ID календаря, в который будут добавляться события.
// Чтобы использовать календарь по умолчанию, оставьте 'default'.
// Чтобы найти ID другого календаря: зайдите в его настройки, раздел
"Интеграция календаря".
CALENDAR_ID: 'default',

// Часовой пояс для корректного определения дат.
// Список часовых поясов:
https://en.wikipedia.org/wiki/List\_of\_tz\_database\_time\_zones
TIME_ZONE: 'Europe/Yekaterinburg',

// За сколько дней вперед создавать события о днях рождения.
// Например, 31 день — события будут созданы на ближайший месяц.
DAYS_AHEAD_TO_CREATE_EVENTS: 31,

// Местоположение для событий в календаре (необязательно).
EVENT_LOCATION: 'Пермь',

// Настройки уведомлений (в минутах до начала события).
// 0 = в момент начала (в 00:00), 900 = за 15 часов (в 9:00 предыдущего дня).
REMINDER_MINUTES: [0, 900],

// Настройки для файла логов на Google Диске.
LOG_FILE_SETTINGS: {
ENABLED: true, // Включить или выключить сохранение логов в файл
FILE_NAME_SUFFIX: '_BirthdayLogs.txt' // Суффикс для имени файла лога
}
};

// --- ГЛАВНАЯ ФУНКЦИЯ ---

/**
* Основная функция. Получает контакты и создает события в календаре для
предстоящих дней рождения.
*/
function createBirthdayEvents() {
const calendar = getCalendar();
if (!calendar) return;

const {
startDate,
endDate
} = getDateRange();
logScriptRunDates(startDate, endDate);

```

ia.panorama  
12 постов

[Посмотреть весь топ](#)



**Лучшие посты  
недели**



Рассылка Пикабу:  
отправляем самые  
рейтинговые материалы за 7  
дней 🔥

Укажи

**Подписаться**

Нажимая «Подписаться»,  
я даю согласие на [обработку](#)  
[данных](#) и [условия почтовых](#)  
[рассылок](#).

Реклама

Помощь	Правила
Кодекс Пикабу	соцсети
Команда	О
Пикабу	рекомендация
Моб.	х
приложение	О компании

Промокоды Биг Гик  
Промокоды Lamoda  
Промокоды МВидео  
Промокоды Яндекс Маркет  
Промокоды Пятерочка  
Промокоды Aroma Butik  
Промокоды Яндекс  
Путешествия  
Промокоды Яндекс Еда  
Постила  
Футбол сегодня





```

const contacts = getAllContactsWithBirthdays();
if (contacts.length === 0) {
  Logger.log('Не найдено контактов с указанной датой рождения.');
```

return;

```

}

Logger.log(`Найдено ${contacts.length} контактов с датой рождения.
Начинаем обработку...`);

let eventsCreated = 0;
contacts.forEach(contact => {
  const birthdayInfo = getBirthdayInfo(contact, startDate.getFullYear());

  if (birthdayInfo.nextBirthday >= startDate && birthdayInfo.nextBirthday <=
endDate) {
    Logger.log(`-> День рождения "${birthdayInfo.name}" (${birthdayInfo.dateString})
попадает в диапазон.`);
    deleteExistingEvents(calendar, birthdayInfo.name, birthdayInfo.nextBirthday);
    createCalendarEvent(calendar, birthdayInfo);
    eventsCreated++;
  }
});

Logger.log(`Обработка завершена. Создано/обновлено
событий: ${eventsCreated}.`);
if (CONFIG.LOG_FILE_SETTINGS.ENABLED) {
  saveLogToDrive();
}
}

// --- ВСПОМОГАТЕЛЬНЫЕ ФУНКЦИИ ---

/**
 * Получает объект календаря по ID из настроек.
 * @Returns {CalendarApp.Calendar|null} Объект календаря или null в
случае ошибки.
 */
function getCalendar() {
  try {
    const calId = CONFIG.CALENDAR_ID;
    const calendar = calId === 'default' ?
CalendarApp.getDefaultCalendar() :
CalendarApp.getCalendarById(calId);

    if (!calendar) {
      throw new Error(`Календарь с ID "${calId}" не найден.`);
    }

    Logger.log(`Используется календарь: "${calendar.getName()}");
    return calendar;
  } catch (e) {
```



```

    Logger.log(`Ошибка при получении календаря: ${e.message}`);
    return null;
  }
}

/**
 * Определяет диапазон дат для поиска дней рождения.
 * @Returns {{startDate: Date, endDate: Date}} Объект с начальной и
 * конечной датами.
 */
function getDateRange() {
  const startDate = new Date();
  const endDate = new Date(startDate.getTime() +
    CONFIG.DAYS_AHEAD_TO_CREATE_EVENTS * 24 * 60 * 60 * 1000);
  return {
    startDate,
    endDate
  };
}

/**
 * Логирует диапазон дат выполнения скрипта.
 * @Param {Date} startDate Начальная дата.
 * @Param {Date} endDate Конечная дата.
 */
function logScriptRunDates(startDate, endDate) {
  Logger.log(`Скрипт запущен. Поиск дней рождения в диапазоне:`);
  Logger.log(`C: ${Utilities.formatDate(startDate, CONFIG.TIME_ZONE,
    'dd.MM.yyyy HH:mm')}`);
  Logger.log(`По: ${Utilities.formatDate(endDate, CONFIG.TIME_ZONE,
    'dd.MM.yyyy HH:mm')}`);
}

/**
 * Получает все контакты пользователя, у которых указана дата рождения,
 * используя People API.
 * @Returns {Array<Object>} Массив объектов контактов.
 */
function getAllContactsWithBirthdays() {
  const allContacts = [];
  let pageToken = null;

  try {
    do {
      const response = People.People.Connections.list('people/me', {
        personFields: 'names,birthdays,phoneNumbers',
        pageSize: 1000,
        pageToken: pageToken
      });
    }
  }

```



```

if (response.connections && response.connections.length > 0) {
  const contactsWithBirthdays = response.connections.filter(person =>
    person.birthdays && person.birthdays.some(b => b.date)
  );
  allContacts.push(...contactsWithBirthdays);
}
pageToken = response.nextPageToken;
} while (pageToken);

} catch (e) {
  Logger.log('Не удалось получить контакты через People API: ${e.message}');
  // Можно добавить отправку уведомления по email в случае
  критической ошибки
  // MailApp.sendEmail('your-email@example.com', 'Ошибка в скрипте дней
  рождения', e.message);
}
return allContacts;
}

/**
 * Извлекает и форматирует информацию о дне рождения контакта.
 * @Param {Object} person Объект контакта из People API.
 * @Param {number} currentYear Текущий год.
 * @Returns {Object} Объект с информацией о дне рождения.
 */
function getBirthdayInfo(person, currentYear) {
  const name = person.names && person.names.length > 0 ?
    person.names[0].displayName : '[Имя не указано]';
  const birthdayData = person.birthdays[0].date;
  const {
    day,
    month,
    year
  } = birthdayData;

  const nextBirthday = new Date(currentYear, month - 1, day);
  // Если день рождения в этом году уже прошел, берем следующий год
  if (nextBirthday < new Date(new Date().setHours(0, 0, 0, 0))) {
    nextBirthday.setFullYear(currentYear + 1);
  }

  const age = year ? nextBirthday.getFullYear() - year : null;
  const ageText = age ? `${age} ${getAgePostfix(age)}` : '';
  const mobilePhone = person.phoneNumbers ? person.phoneNumbers.find(p =>
    p.type === 'mobile') : null;

  return {
    name,
    year,
    age,

```



```

ageText,
nextBirthday,
mobilePhone: mobilePhone ? mobilePhone.value : null,
dateString: `${day}.${month}${year ? '.' + year : ''}`
};
}

/**
 * Удаляет старые события для этого же контакта на ту же дату.
 * @Param {CalendarApp.Calendar} calendar Объект календаря.
 * @Param {string} contactName Имя контакта.
 * @Param {Date} eventDate Дата события.
 */
function deleteExistingEvents(calendar, contactName, eventDate) {
  try {
    const existingEvents = calendar.getEvents(eventDate, new
    Date(eventDate.getTime() + 1), {
      search: contactName
    });
    if (existingEvents.length > 0) {
      Logger.log(` Удаление ${existingEvents.length} старых событий
      для "${contactName}" ...`);
      existingEvents.forEach(event => event.deleteEvent());
    }
  } catch (e) {
    Logger.log(` Ошибка при удалении старых событий: ${e.message}`);
  }
}

/**
 * Создает событие в календаре.
 * @Param {CalendarApp.Calendar} calendar Объект календаря.
 * @Param {Object} birthdayInfo Информация о дне рождения.
 */
function createCalendarEvent(calendar, birthdayInfo) {
  const eventTitle = `${birthdayInfo.name} — день рождения${birthdayInfo.ageText ?
  ', ' + birthdayInfo.ageText : ''}`;
  let description = `Сегодня ${birthdayInfo.name} празднует день
  рождения${birthdayInfo.ageText ? ` — ${birthdayInfo.ageText}` : ''}!\n\nС Днём
  Рождения! 🎂🎁😊🎉👏👤`;
  if (birthdayInfo.mobilePhone) {
    description += `\n\n📞 ${birthdayInfo.mobilePhone}`;
  }

  try {
    const event = calendar.createAllDayEvent(eventTitle, birthdayInfo.nextBirthday, {
      description: description,
      location: CONFIG.EVENT_LOCATION
    });
    CONFIG.REMINDER_MINUTES.forEach(minutes =>

```



```

event.addPopupReminder(minutes));
Logger.log( ✓ Событие "${eventTitle}" успешно создано.);
} catch (e) {
Logger.log( ✗ Не удалось создать событие для
"${birthdayInfo.name}": ${e.message});
}
}

/**
 * Возвращает правильное окончание для возраста ("год", "года", "лет").
 * @Param {number} age Возраст.
 * @Returns {string} Слово с правильным окончанием.
 */
function getAgePostfix(age) {
const lastDigit = age % 10;
const lastTwoDigits = age % 100;

if (lastTwoDigits >= 11 && lastTwoDigits <= 19) {
return 'лет';
}
if (lastDigit === 1) {
return 'год';
}
if (lastDigit >= 2 && lastDigit <= 4) {
return 'года';
}
return 'лет';
}

// — ФУНКЦИИ УПРАВЛЕНИЯ ТРИГГЕРАМИ И ЛОГАМИ —

/**
 * Создает или обновляет триггер для ежемесячного автоматического
запуска скрипта.
 * Запускается 1-го числа каждого месяца. Рекомендуется запустить эту
функцию один раз вручную.
 */
function setupMonthlyTrigger() {
const functionName = 'createBirthdayEvents';
// Удаляем все предыдущие триггеры для этого проекта, чтобы
избежать дублирования
ScriptApp.getProjectTriggers().forEach(trigger => ScriptApp.deleteTrigger(trigger));

// Создаем новый триггер, который будет запускать скрипт 1-го числа
каждого месяца в 1-2 часа ночи
ScriptApp.newTrigger(functionName)
.timeBased()
.onMonthDay(1) // Запускать в 1-й день месяца
.atHour(1) // Указываем час запуска

```



```

.create();

Logger.log(`Триггер для функции "${functionName}" успешно создан. Он будет
запускаться ежемесячно, 1-го числа.`);
}

/**
 * Сохраняет журнал выполнения (логи) в текстовый файл на Google Диске.
 */
function saveLogToDrive() {
  try {
    const scriptFile = DriveApp.getFileById(ScriptApp.getScriptId());
    const scriptName = scriptFile.getName();
    const parentFolder = scriptFile.getParents().next() || DriveApp.getRootFolder();
    const logFileName =
`${scriptName}${CONFIG.LOG_FILE_SETTINGS.FILE_NAME_SUFFIX}`;

    // Удаляем старый файл лога, если он существует
    const oldLogs = parentFolder.getFilesByName(logFileName);
    if (oldLogs.hasNext()) {
      oldLogs.next().setTrashed(true);
    }

    // Создаем новый файл с текущими логами
    parentFolder.createFile(logFileName, Logger.getLog());
    Logger.log(`Логи сохранены в файл: "${logFileName}" в папке
"${parentFolder.getName()}"`);
  } catch (e) {
    Logger.log(`Ошибка при сохранении лога на Диск: ${e.message}`);
  }
}

// --- ТЕСТОВАЯ ФУНКЦИЯ ---

/**
 * Тестовая функция. Находит и выводит в лог все контакты, у которых есть
дата рождения.
 * Ничего не создает и не изменяет.
 */
function test_listAllBirthdays() {
  Logger.log(`--- Начало теста: Поиск всех контактов с днями рождения ---`);
  const contacts = getAllContactsWithBirthdays();

  if (contacts.length > 0) {
    contacts.forEach(person => {
      const name = person.names && person.names.length > 0 ?
person.names[0].displayName : '[Имя не указано]';
      const bday = person.birthdays[0].date;
      Logger.log(`Найден контакт: ${name} (ДР: ${bday.day || '?'} ${bday.month ||

```



```
'?').${bday.year || '????')});  
});  
Logger.log(`--- Тест завершен. Всего найдено контактов с днем рождения:  
${contacts.length} ---`);  
} else {  
Logger.log("--- Тест завершен. Контакты с днями рождения не найдены. ---");  
}  
}
```

1. Обратите внимание на блок CONFIG в самом верху. Вам не нужно разбираться в логике скрипта. Хотите уведомление не в 00:00, а за 15 часов? Поменяйте цифру в REMINDER\_MINUTES. Живете не на Урале? Впишите свой TIME\_ZONE. Это как заполнить анкету — всё интуитивно понятно.
2. При первом запуске нажмите кнопку «Выполнить» (треугольник Play) для функции createBirthdayEvents. И тут Google попытается вас защитить.

Вы увидите грозное окно: «Приложение не проверено». Не пугайтесь. Это стандартная процедура: Google предупреждает, что автор скрипта — неизвестный разработчик (то есть вы сами). Чтобы продолжить:

- Нажмите Проверить разрешения.
- Выберите аккаунт.
- Нажмите Разрешить, чтобы дать скрипту доступ к вашим контактам и календарю.

Всё. Скрипт получил доступ к «топливу» и готов работать. Вы в любое время можете посмотреть список выданных вами [разрешений на специальной странице](#) и в один клик их отозвать.

## Немного технических подробностей

Выбор People API не случаен: старый Contacts API устарел и перестал работать (API контактов [был отключен 19 января 2022 г.](#)). Новый интерфейс гарантирует, что интеграция не «отвалится» при очередном обновлении Google.

За гигиену календаря отвечает функция deleteExistingEvents. Она предотвращает создание дублей: перед записью скрипт проверяет и удаляет старую метку на этот день.

Для контроля добавлено логирование на Google Диск. Полный отчет о работе сохраняется в текстовый файл: так проще найти ошибку в данных контакта, не открывая консоль разработчика.

## Автоматизация: «поставь и забудь»

Чтобы скрипт работал как настоящий ассистент, нужно один раз настроить триггер. Это автоматически запустит обновление событий без вашего участия.

Запустите функцию setupMonthlyTrigger или в интерфейсе Google Apps Script откройте «Триггеры», создайте новый и выберите функцию createBirthdayEvents.



Затем укажите запуск 1-го числа каждого месяца. Календарь всегда заполнен на ближайшие 30 дней, без лишних ежедневных перезапусков и нагрузки на аккаунт.

Триггеры

### Этический момент

Напоминание о возрасте — это не попытка «подсветить» лишние цифры, а всего лишь способ не попасть в неловкую ситуацию. Скрипт показывает возраст только вам, в вашем календаре, и нигде больше не всплывает. Использовать эту информацию или нет — целиком ваш выбор. Кому-то приятно получить поздравление «с круглой датой», а кому-то достаточно тёплых слов без упоминания цифр.

### Заключение

Автоматизация напоминаний о днях рождения — это маленькое улучшение, которое экономит массу времени и снижает нагрузку на мозг.

Вместо ручных подсчётов и переходов по меню вы получаете готовую, аккуратную запись с возрастом и ключевыми данными контакта.

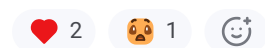
Благодаря People API решение работает стабильно, не зависит от устаревших сервисов и легко адаптируется под ваши привычки через простой CONFIG-блок. Один раз настроили — и дальше календарь сам делает рутину за вас.

**Автор:** Михаил Шардин

 [Моя онлайн-визитка](#)

 [Telegram «Умный Дом Инвестора»](#)

25 ноября 2025





8



tulpanovka.ru

**Купить таунхаус в Сочи. Дома в ЖК «Тюльпановка»**

5,0 ★ Рейтинг организации ⓘ

[Подробнее](#)

foton-mbrus.ru

**Пикап FOTON TUNLAND V9**

FOTON TUNLAND V создан для тех, кто привык побеждать! Оставьте заявку!

[Подробнее](#)**MS, Libreoffice & Google docs**

762 поста • 14.9K подписчика

[Добавить пост](#)[Подписаться](#)**Правила сообщества**


1. Не нарушать [правила Пикабу](#)
2. Публиковать посты соответствующие тематике сообщества
3. Проявлять уважение к пользователям...

[Подробнее](#) ✓[Все комментарии](#) [Автора](#)[Раскрыть 8 комментариев](#)Чтобы оставить комментарий, необходимо [зарегистрироваться](#) или [войти](#)

Реклама


Реклама

Реклама



**3000 разных курсов на одной платформе**

Сравнивай, читай отзывы и выбирай тот, который подходит именно тебе





■ ————— ■ —————

—————

—————

—————

—————

—————

—————

● ————— ■ —————

—————

—————

—————

—————

—————

—————

● ————— ■ —————

—————

—————

—————

—————

—————

—————

