



Girls Who Code At Home

उल्का पकड़ो गेम: भाग 1
योजना बनाना + p5.js से परिचय

गतिविधि अवलोकन

इस प्रोजेक्ट में आप **p5.js** नामक एक जावास्क्रिप्ट लाइब्रेरी का उपयोग करके एक एकत्रण गेम को प्रोग्राम करना सीखेंगे; यह लाइब्रेरी विशेष रूप से कलाकारों और डिज़ाइनर्स के लिए बनाई गई है। इस प्रोजेक्ट के **भाग 1** में, आप जानेंगे कि गेम के विभिन्न भाग साथ मिलकर एक सिस्टम के रूप में कैसे कार्य करते हैं, और आप p5.js की बुनियादी बातें जानेंगे। आप फंक्शन्स का उपयोग करके रेखा-चित्र बनाएंगे और प्रोग्राम फ़्लो यानि प्रोग्राम में कोड कैसे निष्पादित होता है इस बारे में और जानेंगे।



सीखने के लक्ष्य

इस गतिविधि को पूरा कर लेने पर आप निम्नलिखित कर सकेंगे...

- ❑ गेम के भागों की पहचान करना और समझाना कि प्रोग्राम में वे किस प्रकार साथ मिलकर एक सिस्टम के रूप में कार्य करते हैं
- ❑ p5.js के पीछे की प्रेरणा का वर्णन करना और परिवेश में यहां-वहां जाना।
- ❑ संबंधित शब्दों, जैसे कंडीशनल्स और कंट्रोल फ़्लो, का उपयोग करते हुए प्रोग्राम के फ़्लो (प्रवाह) का वर्णन करना।

सामग्रियां

- [p5.js ऑनलाइन एडिटर](#)
- [उल्का पकड़ो गेम सैंपल प्रोजेक्ट](#)
- [उल्का पकड़ो गेम भाग 1 संदर्भ मार्गदर्शिका](#)

पूर्व ज्ञान

इस प्रोजेक्ट से शुरुआत करने से पहले, हमारी सलाह है कि आप:

- ❑ अपने शब्दों में यह स्पष्ट कर सकते हों कि चर राशि या [variable](#) क्या होता है और यह बता सकते हों कि किसी प्रोग्राम में उनका उपयोग कैसे किया जा सकता है।
- ❑ अपने शब्दों में यह स्पष्ट कर सकते हों कि सशर्त कथन या कंडीशनल स्टेटमेंट ([conditional statement](#)) क्या होता है और यह बता सकते हों कि किसी प्रोग्राम में उनका उपयोग कैसे किया जा सकता है।

वुमन इन टेक स्पॉटलाइट: कैसी टारकजिन (Cassie Tarakajian)



तस्वीर स्रोत: [NYU Tisch](#)

कैसी टारकजिन (Cassie Tarakajian) एक सॉफ्टवेयर डेवलपर, हार्डवेयर इंजीनियर, रचनाशील प्रौद्योगिकीविद, संगीतज्ञ, और शिक्षक हैं। कैसी (Cassie) स्वयं को [नॉन-बाइनरी \(अन्य लिंग का\)](#) मानती हैं और लिंगहीन सर्वनामों का उपयोग करती हैं। कैसी (Cassie) ने कोड करना सबसे पहले कॉलेज में सीखा था, जहां उन्होंने एक इंटीरियर डिजाइन टू [जावा](#) कक्षा में भाग लिया था। उन्हें याद है कि बेहद साधारण, टेक्स्ट-आधारित काय को पूरा करने के लिए उन्होंने एक बेहद साधारण टेक्स्ट एडिटर का उपयोग करके कोडिंग सीखी थी। बाद में चलकर, कैसी (Cassie) से [p5.js](#) नामक एक ओपन-सोर्स प्रोजेक्ट में योगदान करने को कहा गया; यह एक जावास्क्रिप्ट लाइब्रेरी थी जो प्रोसेसिंग प्लेटफॉर्म के आधार पर परस्पर व्यवहारी कला और ध्वनियां बनाने के लिए है। p5.js वेब एडिटर की रचयिता और मुख्य मेन्टेनर होने के नाते, उन्होंने एक ऐसा परिवेश बनाया जहां लोग सीधे कोड लिख कर (..) में चला सकते हैं

ब्राउज़र में ही कोड लिख और चला सकते थे, जिससे नौसिखियों के लिए इसका उपयोग आसान हो गया। यह प्लेटफॉर्म समुदाय के उन लोगों के लिए भी पूरी तरह सुगम्य है जो दृष्टिबाधित हैं, और यह प्लेटफॉर्म स्क्रीन रीडर्स और हाई कॉन्ट्रास्ट व्यू के साथ संगतता प्रदान करता है।

p5.js में कैसी (Cassie) के कार्य के साथ-साथ, वे [साइक्लिंग '74 \(Cycling '74\)](#) में इंजीनियर हैं, [गर्लफ्रेंड्स लैब \(Girlfriends Lab\)](#) की संस्थापक हैं, और न्यू यॉर्क यूनिवर्सिटी (NYU) के [टिश स्कूल ऑफ़ द आर्ट्स \(Tisch School of the Arts\)](#) में सहायक प्राध्यापक हैं। कैसी (Cassie) इन सभी भूमिकाओं को निरंतर अच्छी तरह निभाते हुए यह सिद्ध कर रही हैं कि प्रोग्रामर की छवि मात्र एक वर्णन तक सीमित नहीं है।

कैसी (Cassie) और p5.js के साथ उनके कार्य के बारे में और जानने के लिए यह [वीडियो](#) (6:00 तक) देखें। कैसी (Cassie) और उनके कार्य के बारे में और जानना चाहते हैं? उनकी [व्यक्तिगत वेबसाइट](#) देखें। p5.js से संबंधित उनके कार्य और उनकी शुरुआत कैसे हुई इस बारे में यह [लेख](#) पढ़ें।

झलक

एक कंप्यूटर वैज्ञानिक होना, कोडिंग में बेहतरीन होने की तुलना में अधिक है। इस बात के बारे में सोचने में थोड़ा समय बिताएं कि कैसे थेरेसा और उनका काम उन शक्तियों से संबंधित है जिन पर महान कंप्यूटर वैज्ञानिक - बहादुरी, लचीलेपन, रचनात्मकता और उद्देश्य के निर्माण के दौरान ध्यान केंद्रित करते हैं।



प्रयोजन

p5.js की रचना करते समय कैसी (Cassie) के लिए एक ऐसा परिवेश बनाना महत्वपूर्ण था जो न केवल कोड करना सीखना आसान बनाए, बल्कि जो विभिन्न प्रकार की आवश्यकताओं के लिए सुगम भी हो। स्क्रीन रीडर्स के साथ संगत होना और कस्टमाइज़ की जा सकने वाली सेटिंग्स प्रदान करना, p5.js में सुगम्यता का एक मुख्य घटक है। आपके विचार में कैसी (Cassie) के लिए यह महत्वपूर्ण क्यों था कि वे “सबके लिए” वेब एडिटर की रचना करें?

परिवार के किसी सदस्य या मित्र के साथ अपनी प्रतिक्रियाएँ साझा करें। चर्चा में शामिल होने हेतु दूसरों को कैट के बारे में अधिक पढ़ने के लिए प्रोत्साहित करें!

चरण 1: उल्का पकड़ी गेम को जानें (10-15 मिनट)

गेम के भागों से मिलें (2 मिनट)

सभी (या अधिकतर) गेम्स में छः भाग होते हैं: एक लक्ष्य, एक चुनौती, मूल यांत्रिकी, घटक, नियम, और स्थान। ये भाग साथ मिलकर एक सिस्टम के रूप में कार्य करते हैं जो शामिल लोगों के बीच खेल की रचना करता है। गेम डिज़ाइनर होने के नाते यह ज़रूरी है कि अपने गेम की प्रोग्रामिंग करने के लिए आप समझते हों कि सभी भाग साथ मिलकर एक सिस्टम के रूप में कैसे कार्य करते हैं।



- **लक्ष्य:** खिलाड़ी या टीम को गेम जीतने के लिए क्या करना होता है?
- **चुनौती:** लक्ष्य तक पहुंचने के खिलाड़ी के रास्ते में कौन-कौनसी बाधाएं होती हैं?
- **मूल यांत्रिकी:** गेम के खेल को संचालित करने के लिए खिलाड़ी कौनसी मूल क्रियाएं या संचलन करता है?
- **पुर्ज:** खेल की सामग्री किन भागों से मिलकर बनी होती है?
- **नियम:** किन संबंधों से यह तय होता है कि खिलाड़ी गेम में क्या कर सकता है और क्या नहीं?
- **अंतरिक्ष:** गेम कहां घटित होता है?

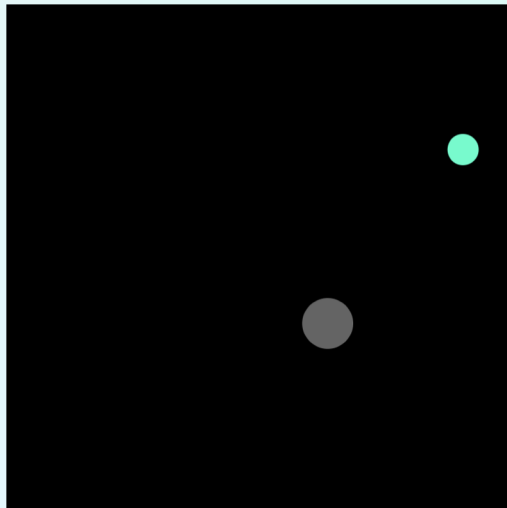
उदाहरण

टिक टैक टो का उदाहरण

- **लक्ष्य:** सबसे पहले तीन एक जैसे निशानों की कतार बनाएं
- **चुनौती:** आपको नहीं पता कि आपका विरोधी अपना निशान कहां रखेगा
- **मूल यांत्रिकी:** ब्लॉक करना और लेखन
- **पुर्ज:** लेखन पात्र, कागज़, खिलाड़ी, X और O
- **नियम:** इसमें 2 खिलाड़ी होते हैं। हर खिलाड़ी बारी-बारी से अपना निशान तब तक लिखता है जब तक ग्रिड पूरी न भर जाए या कोई एक खिलाड़ी आड़ी, खड़ी या तिरछी कतार में तीन निशान न लिख ले।
- **स्थान:** कागज़ पर 3x3 का ग्रिड

गेम खेलें (1 मिनट)

गेम डिज़ाइनर के लिए गेम खेलना, अपने कौशलों का अभ्यास करने का सबसे अच्छा तरीका होता है! चलिए वह गेम खेल कर देखते हैं जो हम बनाएंगे। गेम को लगभग 30 सेकंड तक खेलने के लिए इस [लिंक](#) पर क्लिक करें। खेल के दौरान, इस गेम के भागों की पहचान करने की कोशिश करें।



हम नए प्रोग्रामिंग कौशल सीखने के लिए उल्काएं पकड़ने के एक गेम की योजना बनाएंगे और उस गेम की रचना करेंगे, पर विस्तार अनुभागों में आपको उसमें आवश्यकतानुसार बदलने का मौका भी मिलेगा

चरण 1: उल्का पकड़ो गेम को जानें (जारी)

गेम के भागों की पहचान करें (5-10 मिनट)

इस चरण में हमारा लक्ष्य बड़ी समस्याओं को छोटे-छोटे टुकड़ों में तोड़ कर यह समझना है कि हमारे उल्का पकड़ो गेम का सिस्टम किस प्रकार कार्य करता है। इस प्रक्रिया को **वियोजन (डीकम्पोज़िशन)** कहते हैं।

मान लें कि आपको बच्चों के एक समूह के लिए 100 पिज़्ज़ा बनाने हैं। यह बहुत बड़ा काम है, पर यदि हम इसे छोटे-छोटे चरणों और उप-समस्याओं में तोड़ दें, तो यह कहीं अधिक आसान हो जाएगा। उदाहरण के लिए, सबसे पहले सारा आटा गूंथ लें, सारी सॉस पका लें, एक बार में पांच पिज़्ज़ा तैयार करें, और उन्हें बैच में बांट कर पकाएं।



आप किसी भी जटिल समस्या को कई सारे तरीकों से अपेक्षाकृत सरल भागों में तोड़ सकते हैं। चूंकि हमारे पास खेलने के लिए गेम पहले से है, अतः हम **रिवर्स इंजीनियरिंग** का तरीका अपनाएंगे। इसका अर्थ है कि शून्य से गेम बनाने की कोशिश करने की बजाय, हम तैयार उत्पाद के भागों को अलग-अलग करके पता करेंगे कि उसे कैसे बनाया गया था। सबसे पहले, हम उल्का पकड़ो गेम के सभी भागों को परिभाषित करेंगे, और फिर उनका उपयोग करके स्पूडोकोड (छद्म कोड) लिखेंगे। नीचे दिए गए स्थान में उल्का पकड़ो गेम के भागों को छोटे-छोटे भागों में तोड़ने की कोशिश करें। अपने विचारों को **संदर्भ मार्गदर्शिका** से जांचना न भूलें।

उल्का पकड़ो गेम के भाग

- अभी पिछले चरण में आपने जो उल्का पकड़ो गेम खेला था, उसके **लक्ष्य** का वर्णन करें। खिलाड़ी या टीम को गेम जीतने के लिए क्या करना होता है?
- उल्का पकड़ो गेम के **घटक**ों में उल्का, कैचर, दीवारें और खिलाड़ी शामिल हैं। हर घटक के अपने अलग गुण हैं (जैसे आकार, रंग, आकृति आदि) और अलग क्रियाएं हैं (यानि वे चीज़ें जो वह करता है - वे क्रियाएं जो आप उस घटक से जोड़ कर देखते हैं) और ये गुण व क्रियाएं गेम के सिस्टम में योगदान देते हैं। उदाहरण के लिए, उल्का का गुण होगा उसका गोल होना, और उसकी क्रिया होगी स्क्रीन के ऊपरी भाग से नीचे की ओर गिरना। **नीचे दी गई तालिका में बताए गए हर घटक के गुणों और क्रियाओं के बारे में 2-3 मिनट सोचें।**

पुर्जा	गुण	कार्रवाई
खेल की आवश्यक वस्तुएं क्या हैं?	घटक के गुण या विशेषताएं क्या हैं? (जैसे आकार, रंग, आकृति, आदि)	वह क्या करता है? आप उसके साथ कौनसी क्रियाएं जोड़ते हैं?



चरण 1: उत्का पकड़ी गेम को जानें (जारी)

- गेम के **स्थान** का वर्णन करें। वह कहां घटित होता है? (ध्यान दें कि कभी-कभी स्थान, एक से अधिक चीज़ें हो सकता है। उदाहरण के लिए शतरंज, बिसात पर खेला जाता है, पर वह किसी कमरे, पार्क या कैफ़ेटेरिया में भी खेला जा रहा होता है।)
- **चुनौती** को परिभाषित करें। लक्ष्य तक पहुंचने के खिलाड़ी के रास्ते में कौन-कौनसी बाधाएं होती हैं?
- गेम की **मूल यांत्रिकी** का वर्णन करें। गेम को खेलने के लिए खिलाड़ी कौनसी मूल क्रियाएं या संचलन करता है? गेम के खेल को संचालित करने के लिए खिलाड़ी कौनसी मूल क्रियाएं या संचलन करता है?
- गेम के **नियमों** की सूची तैयार करें। नियमों से तय होता है कि हम हमारे गेम में क्या कर सकते हैं और क्या नहीं। वे खिलाड़ियों, घटकों, स्थान, आदि पर लागू किए जा सकते हैं।



अपने विचारों को पृष्ठ 3 पर संदर्भ मार्गदर्शिका से जांचना न भूलें।

चरण 2: अपने गेम का स्कूडोकोड (छद्म कोड) लिखें (5-10 मिनट)



इस चरण में, किसी कागज़ पर या कंप्यूटर में अपने प्रोग्राम के उच्च स्तरीय निर्देश लिखने की कोशिश करें। इसे **स्कूडोकोड (छद्म कोड)** कहते हैं। स्कूडोकोड या छद्म कोड साधारण भाषा में इस बात का वर्णन होता है कि आपका कोड क्या करेगा। इससे आपको अपने प्रोग्राम का फ़्लो (प्रवाह) और लॉजिक (तर्क) सोचने में मदद मिलती है, जिससे आप वे कदम तय कर पाते हैं जो आपको अपना प्रोग्राम लिखने के लिए उठाने होंगे। स्कूडोकोड (छद्म कोड) विशिष्ट कोड वाक्य-विन्यास का उपयोग किए बिना काफ़ी हद तक कोड जैसा दिख सकता है। उदाहरण के लिए, आप if या ऐसे अन्य मूल मुख्य-शब्दों (कीवर्ड्स) का उपयोग कर सकते हैं जो सभी प्रोग्रामिंग भाषाओं में पाए जाते हैं। शुरुआत हमेशा स्कूडोकोड (छद्म कोड) से करें!

शुरुआत करने में आपकी मदद के लिए हमने कुछ चीज़ें पहले ही लिख दी हैं। आपको क्या-क्या चीज़ें शामिल करनी हैं यह तय करना आसान बनाने के लिए, आपको ऊपर बताए गए गेम के भागों का उपयोग भी करना चाहिए। यदि आप कहीं अटक जाएं, तो खुद से गेम के बारे में प्रश्न पूछें। उदाहरण के लिए:

- गेम के आरंभ में क्या घटित होना चाहिए?
- जब गेम खेला जा रहा हो तो उस दौरान क्या घटित होना चाहिए?

अपने उत्तर को अधिक-से-अधिक विशिष्ट बनाने की कोशिश करें, पर यदि आप सब कुछ दर्ज न कर पाएं या यदि आप कुछ ऐसा लिख लें जिसे करना आपको न आता हो, तो चिंता न करें। हर कोई स्कूडोकोड (छद्म कोड) अलग-अलग ढंग से लिखता है! हम प्रोजेक्ट के दौरान इस स्कूडोकोड (छद्म कोड) पर वापस लौटेंगे, इसलिए अपना कार्य सहेजना न भूलें।

```
// शुरुआती स्कूडोकोड (छद्म कोड)  
जो भी चर राशि हों उन्हें घोषित करें
```

```
इसे एक बार करें  
कैनवास का आकार 400 पिक्सल गुणा 400 पिक्सल सेट करें
```

```
इसे हर लूप पर करें  
पृष्ठभूमि का रंग सेट करें  
// हम बाद में इस पर चर्चा करेंगे कि यह setup() की बजाय draw() में क्यों रहता है!  
उल्का बनाएं  
// बाकी की चीज़ें खुद जोड़ने की कोशिश करें!
```

स्क्रीन पर उल्का बना देने के बाद गेम में क्या होता है? यदि आपको याद न आए तो गेम दोबारा खेल कर देखें।

सुझाव: घटक और नियम विशेष रूप से सहायक सिद्ध होंगे!



अपने विचारों को पृष्ठ 4 पर संदर्भ मार्गदर्शिका से जांचना न भूलें।

चरण 2: p5.js से मिलें! (10-15 मिनट)

P5.js (या संक्षेप में सिर्फ p5) की मदद से आप वेब ब्राउज़र्स के लिए परस्पर-व्यवहारी कलाकृतियां बना सकते हैं। यह रचनाशील कोडिंग का एक टूल है - रचनाशील कोडिंग का अर्थ ऐसे प्रोजेक्ट्स से है जो मात्र कार्यक्षमता की बजाय अभिव्यक्ति के लिए कोड का उपयोग करते हैं। P5.js जावास्क्रिप्ट (JavaScript) नामक एक प्रोग्रामिंग भाषा की एक लाइब्रेरी है जो आपको वेब पर परस्पर-व्यवहार (इंटरैक्टिविटी) जोड़ने की सुविधा देती है। लाइब्रेरी होने का यह अर्थ है कि p5 जावास्क्रिप्ट ही है, पर इसके रचयिताओं ने विशेष फंक्शन्स/मैथड्स का एक संकलन (यानि लाइब्रेरी) बना दिया है ताकि आपको हर चीज़ नए सिरे से न बनानी पड़े। चूंकि यह वेब-आधारित है, अतः आप अपनी समस्त रचनाएं आसानी से साझा कर सकते हैं! आप [p5 होमपेज](#) पर इसकी जड़ों और इसके समुदाय के बारे में और पढ़ सकते हैं। लोगों ने इससे कुछ उदाहरण परियोजना बनाए हैं जिन्हें आप [शोकेस पेज](#) पर देख सकते हैं।



p5.js में p का अर्थ प्रोसेसिंग से है। प्रोसेसिंग (Processing), कलाकारों और डिज़ाइनर्स के लिए बनाई गई एक प्रोग्रामिंग भाषा है जिससे वे अपने प्रोजेक्ट्स में कोड शामिल कर सकते हैं। प्रोसेसिंग को नौसिखियों के लिए बनाया गया था ताकि वे विभिन्न प्रकार का परस्पर-व्यवहारी यानि इंटरैक्टिव मीडिया बना सकें, जैसे एनिमेशन, डेटा विज़ुअलाइज़ेशन, वाद्ययंत्र, गेम्स, और यहां तक कि बड़े पैमाने की स्थापनाएं भी। प्रोसेसिंग के बारे में और जानने के लिए [प्रोसेसिंग फ़ाउंडेशन के होमपेज](#) पर जाएं।

अपना अकाउंट बनाएं (3-5 मिनट)



आप दो तरह से p5.js का उपयोग कर सकते हैं: ऑनलाइन वेब एडिटर से या फिर किसी टेक्स्ट एडिटर और p5.js की एक कॉपी से जिसे आप अपने स्थानीय कंप्यूटर पर डाउनलोड करते हैं। ऑनलाइन एडिटर, p5 के साथ शुरुआत करने का सबसे आसान तरीका है। इसमें आप वेब ब्राउज़र में ही कोड लिख सकते हैं और अपना प्रोग्राम चला सकते हैं। इस ट्यूटोरियल में हम चरणों का संदर्भ लेने और उदाहरणों को स्पष्ट करने के लिए केवल वेब एडिटर का ही उपयोग करेंगे।

वेब एडिटर के साथ शुरुआत करने के लिए, आपको एक अकाउंट बनाना होगा।

- ❑ <https://editor.p5js.org/signup> पर जाएं।
- ❑ **साइन अप करें।** सभी फ़ील्ड्स (यूज़रनेम, ईमेल, पासवर्ड, और पासवर्ड कन्फ़र्मेशन) भरें, और फिर “Sign up” पर क्लिक करें, या फिर आप Google अथवा GitHub से साइन इन करने का विकल्प भी चुन सकते हैं।
- ❑ **अपने ईमेल की पुष्टि करें।** आपको अपने अकाउंट की पुष्टि और सत्यापन के लिए एक ईमेल मिलेगा (यदि वह 3-5 मिनटों में न पहुंचे तो स्पैम फ़ोल्डर जांचें)। लिंक पर क्लिक करें, और फिर अपने चमचमाते नए क्रिडेन्शियल्स (यानि यूज़रनेम और पासवर्ड) से साइन इन करें।
- ❑ **अपने क्रिडेन्शियल्स किसी सुरक्षित स्थान में सहेज लें ताकि आप दोबारा लॉग इन कर सकें।** यदि फिर भी आप अपना पासवर्ड भूल जाएं, तो [लॉग इन \(Log In\)](#) पेज पर जाएं और सबसे नीचे “Reset Your Password” पर क्लिक करें।

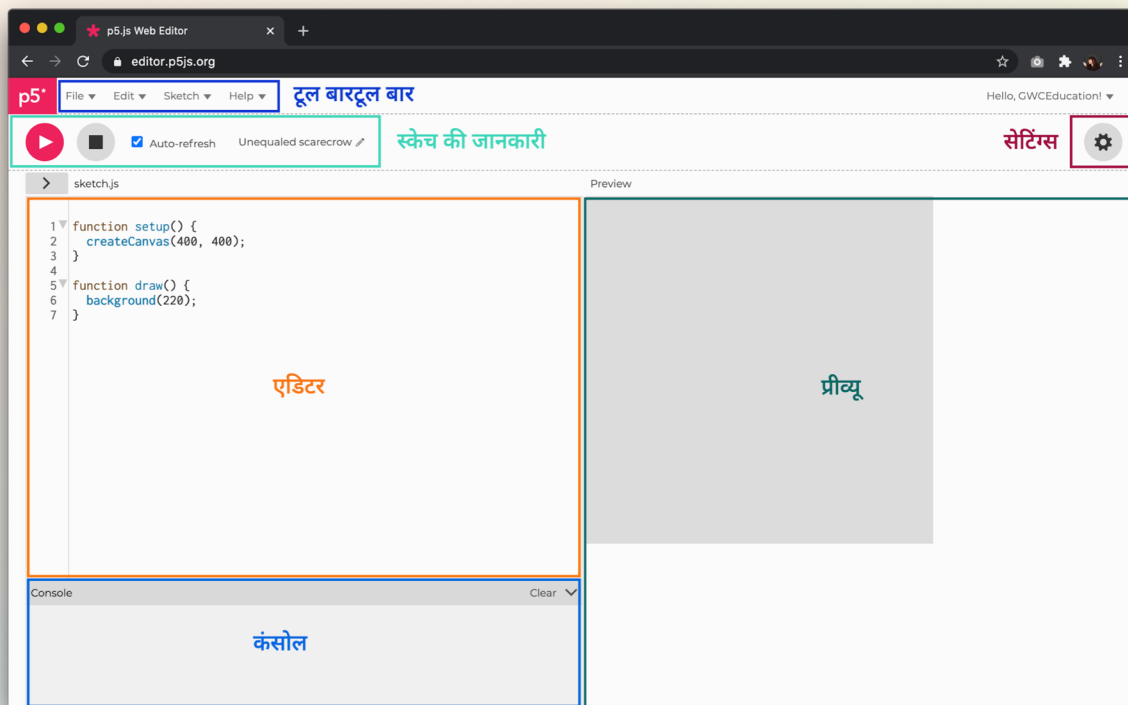
यदि आपका इंटरनेट कनेक्शन अटक-अटक कर चलता हो, या आप स्थानीय एडिटर में काम करना पसंद करते हों, तो आप दूसरा विकल्प उपयोग में ला सकते हैं। यह [गेटिंग स्टार्टेड \(Getting Started\) पेज](#) देखें, यहां आपको अपनी ज़रूरत की सामग्री और लाइब्रेरी डाउनलोड करने के निर्देश मिल जाएंगे। यदि आपको और सहयोग चाहिए हो, तो बेधड़क Google करें!

यदि आप ऑफ़लाइन कार्य कर रहे हैं, तो हो सकता है कि उदाहरण थोड़े अलग दिखें, पर परिणाम समान होंगे।

चरण 3: p5.js से मिलें! (जारी)

परिवेश को जानें (5-8 मिनट)

अब जबकि आपने अपना अकाउंट बना लिया है, चलिए p5 ऑनलाइन एडिटर के इंटरफेस को जानते हैं। यह एक IDE यानि इंटीग्रेटेड डेवलपमेंट एन्वायरन्मेंट (एकीकृत विकास परिवेश) है जो आपको एक ही स्थान पर प्रोग्राम लिखने और चलाने की सुविधा देता है। p5.js में लिखे प्रोग्राम्स को रेखा-चित्र (sketches) कहते हैं। आप इस परिवेश को एक रेखा-चित्रबुक मान सकते हैं जिसमें आपकी ज़रूरत के टूल्स पहले ही दिए हुए हैं!



→ **टूल बार:** पेज में सबसे ऊपर टूलबार होती है।

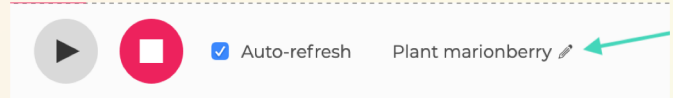
- ◆ **फ़ाइल (File)** मेन्यू में, आप रेखा-चित्र बना सकते हैं, रेखा-चित्र सहेज सकते हैं, रेखा-चित्र की डुप्लीकेट कॉपी बना सकते हैं, रेखा-चित्र को कई फ़ॉर्मेट्स में शेयर कर सकते हैं, रेखा-चित्र फ़ाइल्स डाउनलोड कर सकते हैं, रेखा-चित्र खोल सकते हैं, और उदाहरण खोल सकते हैं। ध्यान दें: इनमें से कुछ विकल्प तब तक नहीं दिखेंगे जब तक आप अपना रेखा-चित्र सहेज नहीं लेते हैं।
- ◆ **एडिट (Edit)** ड्रॉप डाउन मेन्यू से आप अपने कोड को व्यवस्थित कर सकते हैं, अपने रेखा-चित्र में कोई वर्ण या शब्द ढूँढ सकते हैं, और उसमें एक से दूसरे स्थान जा सकते हैं।
- ◆ **रेखा-चित्र (Sketch)** मेन्यू से, आप अपने कोड में फ़ाइल्स या फ़ोल्डर्स जोड़ सकते हैं, और अपने रेखा-चित्र को चला या रोक सकते हैं।
- ◆ आपके लिए सहायक कुंजीपटल (कीबोर्ड) शॉर्टकट्स, p5 संदर्भ पेज की लिंक, और p5 के बारे में और चीज़ें हेल्प (Help) मेन्यू में हमेशा उपलब्ध हैं।

p5.js के कुछ कुंजीपटल (कीबोर्ड) शॉर्टकट्स [यहां](#) देखें।

चरण 3: p5.js से मिलें! (जारी)

- **रेखा-चित्र की जानकारी:** टूलबार के नीचे एक प्ले बटन और एक स्टॉप बटन होता है। प्ले बटन प्रोग्राम को चलाना शुरू करता है। स्टॉप बटन प्रोग्राम को रोक देता है। यदि आप ऑटो-रिफ्रेश ('Auto-refresh') बॉक्स में निशान लगा देते हैं तो प्रोग्राम में बदलाव करने के बाद वह अपने-आप चलता रहेगा, आपको दोबारा प्ले बटन दबाने की ज़रूरत नहीं पड़ेगी।

दायीं ओर आपको अपने रेखा-चित्र का एक पहले से लिखा शीर्षक दिखेगा। अपने रेखा-चित्र का नाम बदलने के लिए, पेंसिल आइकन पर क्लिक करें और नया शीर्षक टाइप करें।



- **सेटिंग्स:** आप रेखा-चित्र की जानकारी के बायीं ओर मौजूद गियर (दांतेदार पहिये) के आइकन पर क्लिक करके सेटिंग्स में पहुंच सकते हैं। यहां आप थीम, टेक्स्ट का आकार, और सुगम्यता (एक्सेसेबिलिटी) सेटिंग्स बदल सकते हैं (हम सुगम्यता के बारे में थोड़ा आगे बात करेंगे)। हमारा सुझाव है कि आप जनरल सेटिंग्स में ऑटोसेव ऑन कर लें।
- **संपादक:** एडिटर वह जगह है जहां आप अपना कोड लिखते हैं। हर लाइन का एक नंबर होता है, जिससे आप हर लाइन को आसानी से संदर्भित कर सकते हैं। नंबर के बगल में मौजूद छोटे-छोटे तीर के निशानों का यह अर्थ है कि आप उन पर क्लिक करके टेक्स्ट को सिकोड़ सकते हैं। उदाहरण के लिए, यदि आपको कई लाइन्स वाले कमेंट्स नहीं देखने तो आप उन्हें सिकोड़ सकते हैं।
- **पूर्वदर्शन:** जब आप प्रोग्राम चलाते हैं तो यह विंडो आपके कोड का परिणाम दिखाती है।
- **कंसोल:** एडिटर के नीचे कंसोल होता है। यह विंडो आपके प्रोग्राम के बारे में जानकारी प्रिंट करती है, जैसे त्रुटि संदेश या फिर वह डेटा जिसे आप प्रोग्राम में एक्सेस करना चाहते हैं, जैसे किसी चर राशि की कीमत।

p5.js में सुगम्यता (एक्सेसेबिलिटी) (2 मिनट)

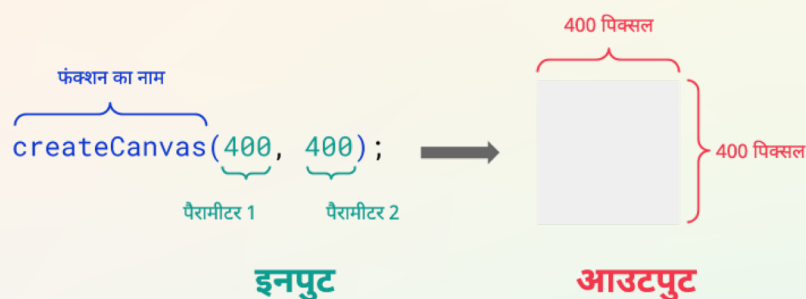


p5 के डेवलपर्स ने दृष्टिबाधित लोगों के लिए एडिटर को सुगम्य बनाने पर विशेष बल दिया है। इन टूल्स का सक्रिय विकास जारी है और ये टूल्स, NYU में होस्ट किए जाने वाले एक कहीं बड़े व निरंतर जारी रिसर्च प्रोजेक्ट का हिस्सा हैं। ऑनलाइन एडिटर वेबसाइट को और खुद एडिटर को भी स्क्रीन रीडर्स द्वारा पढ़ा जा सकता है। सुगम्यता (एक्सेसेबिलिटी) का अधिकांश विकास प्रीव्यू विंडो में आने वाले विजुअल आउटपुट को स्क्रीन रीडर द्वारा पढ़े जाने योग्य बनाने की दिशा में हुआ है। इस विशेषता के उपयोग के बारे में और जानकारी के लिए [p5 वेबसाइट पर यह पेज](#) देखें।

विभिन्न भाषाओं में और विभिन्न प्लेटफॉर्म पर प्रोग्रामिंग करना सीखने के दौरान, आपको सुगम्यता (एक्सेसेबिलिटी) और समावेशन को हमेशा सबसे आगे रखना चाहिए। इतिहास पर नज़र डालें तो डिज़ाइनर्स, इंजीनियर्स, और प्रोग्रामर्स ने सॉफ्टवेयर और हार्डवेयर की रचना के दौरान अशक्तताओं से ग्रस्त लोगों को प्राथमिकता नहीं दी है। चेहरे की पहचान करने वाले एवं अन्य प्रकार के सॉफ्टवेयर के उदय के साथ, यह बात अश्वेत लोगों, महिलाओं और अन्य सीमांत समुदायों पर भी लागू होती है क्योंकि प्रोग्रामर्स के अंतर्निहित पूर्वग्रह (बायस) उनके कोड में भी आ सकते हैं। जागरुकता बढ़ रही है और इसमें बदलाव आने लगा है, पर अभी काफ़ी-कुछ करना बाकी है। समय निकाल कर सुनिश्चित करें कि आपकी रचना का उपयोग हर कोई कर सके!

चरण 4: p5.js के फंक्शन्स को समझें (5-10 मिनट)

प्रोग्राम निर्देशों का एक समूह होता है जिसे आप कंप्यूटर द्वारा पालन किए जाने के लिए बनाते हैं। उन्हीं निर्देशों को बारंबार लिखने की बजाय, हम निर्देशों के छोटे-छोटे टुकड़े बना सकते हैं ताकि हम बाद में उनका दोबारा उपयोग कर सकें। इन टुकड़ों को **फंक्शन** कहते हैं। फंक्शन्स वे कोड की पंक्तियाँ हैं जो कुछ कार्य करती हैं। आप इन्हें क्रियाओं के रूप में देख सकते हैं - वे कुछ *करते* हैं। p5 में, हम अपने प्रोग्राम को फंक्शन्स के रूप में निर्देश देते हैं। आप जिन फंक्शन्स का उपयोग करेंगे उनमें से अधिकांश फंक्शन्स p5.js लाइब्रेरी में परिभाषित हैं (आप अपने स्वयं के फंक्शन्स भी बना सकते हैं, पर इस ट्यूटोरियल में हम इसका तरीका नहीं बताएंगे)।



जब हम हमारे फंक्शन को कॉल करते हैं यानि उसका उपयोग करते हैं, तो प्रोग्राम उसमें मौजूद कोड को चलाता है। उदाहरण के लिए, `createCanvas()` फंक्शन सबसे महत्वपूर्ण फंक्शन्स में से एक है। यह फंक्शन कैनवास एलिमेंट बनाता है जो ग्राफिक्स का चित्रण करता है और रेखा-चित्र का प्रदर्शन करता है। अन्य शब्दों में, यह स्क्रीन का आकार तय करता है। पर हम फंक्शन को यह कैसे बताएं कि हमें कितनी बड़ी स्क्रीन चाहिए? इसके लिए, हम फंक्शन के जरिए कुछ **पैरामीटर्स** पास करते (भेजते) हैं जिससे हमें हमारा मनचाहा आउटपुट मिलता है। पैरामीटर्स कुछ इनपुट वैल्यूज़ होते हैं जिनका उपयोग फंक्शन खुद को संचालित करने के लिए करता है। चलिए `createCanvas()` फंक्शन का वाक्य-विन्यास देखते हैं:

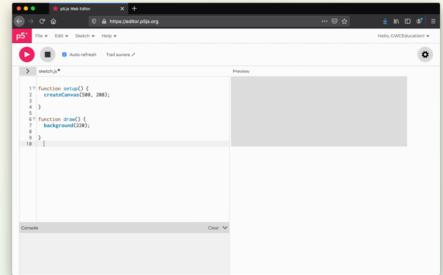
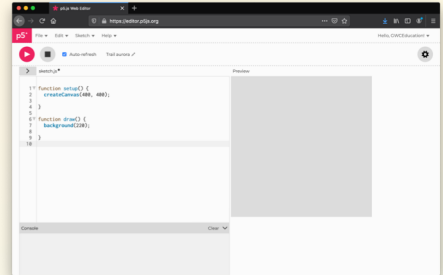
जावास्क्रिप्ट	विवरण
<code>createCanvas(width, height);</code>	<ul style="list-style-type: none">→ createCanvas: फंक्शन का नाम।→ (): हम कोष्ठकों द्वारा हमारे प्रोग्राम को यह बताते हैं कि उसे इस फंक्शन को कॉल करना है। कभी-कभी हम हमारे कोष्ठकों के अंदर फंक्शन के पैरामीटर या इनपुट शामिल कर देते हैं।→ चौड़ाई: पहला पैरामीटर जो कैनवास की चौड़ाई पिक्सल में तय करता है।→ लंबाई: दूसरा पैरामीटर जो कैनवास की ऊंचाई पिक्सल में तय करता है।→ ;: जावास्क्रिप्ट की सभी कोड की पंक्तियों का अंत सेमीकोलन से होना आवश्यक होता है।

चरण 4: p5.js के फंक्शन्स को समझें (जारी)

ये पैरामीटर्स कैनवास की चौड़ाई व ऊंचाई (पिक्सल में) तय करते हैं। डिजिटल स्क्रीन्स बेहद सूक्ष्म खंडों से मिलकर बनी होती हैं और इन ब्लॉकों को पिक्सल कहते हैं। हर पिक्सल, स्क्रीन पर एक अकेला बिंदु दर्शाता है और एक अकेले रंग का होता है। आपको हर p5 रेखा-चित्र में यह फंक्शन शामिल करना होगा।

कैनवास का आकार बदलने के लिए पैरामीटर की कीमतें बदल कर देखें:

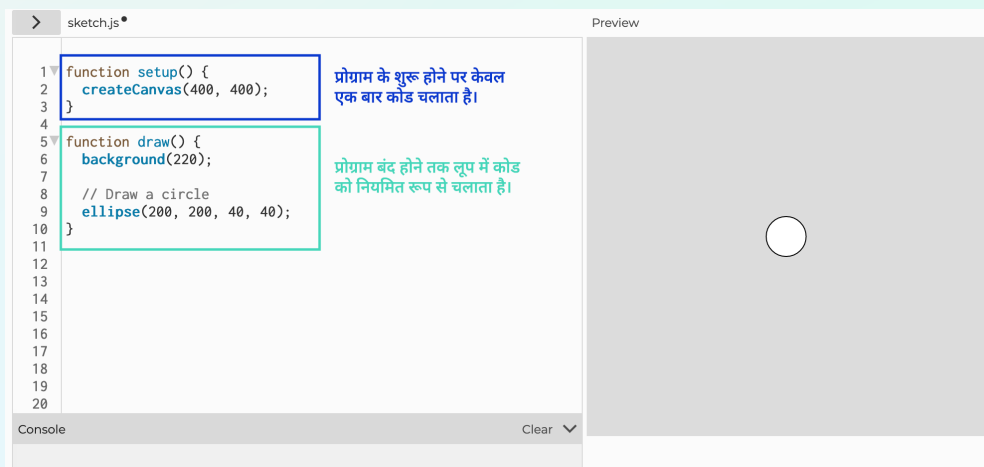
- ❑ **p5 वेब एडिटर खोलें और लॉगइन करें।** शायद आपका ध्यान इस ओर गया होगा कि रेखा-चित्र में शुरुआती कोड पहले से लिखा होता है, जिसमें हमारा दोस्त `createCanvas()` भी है। कैनवास का डिफ़ॉल्ट (पहले से निश्चित) आकार 400 पिक्सल चौड़ा और 400 पिक्सल ऊंचा होता है।
- ❑ **कोड चलाने के लिए प्ले बटन पर क्लिक करें।** कैनवास के आकार पर ध्यान दें।
- ❑ **एक या दोनों कीमत बदलें, और फिर प्ले बटन पर क्लिक करके कोड दोबारा चलाएं।** ऑटो-रिफ्रेश बॉक्स में निशान लगा दें ताकि आपको हर बदलाव के बाद प्ले बटन क्लिक न करना पड़े।



और देखा! प्रीव्यू विंडो में आपके पैरामीटर्स वाले आकार का एक ग्रे रंग का बॉक्स दिखने लगा।

चरण 5: प्रोग्राम के फ़्लो के बारे में जानें (5-10 मिनट)

हम जानते हैं कि हमें हमारे प्रोग्राम को निर्देश कैसे देने हैं, पर हम वे निर्देश रखें कहां? वे कब चलेंगे? क्या निर्देशों का क्रम मायने रखता है? क्या एक फंक्शन के अंदर दूसरा फंक्शन हो सकता है? ये सभी प्रश्न **प्रोग्राम फ़्लो** से जुड़े हैं। इसका अर्थ प्रोग्राम द्वारा आपकी कोड की पंक्तियाँ चलाने के क्रम से है। p5.js में, प्रोग्राम प्रत्येक कोड की पंक्तियाँ क्रम में चलाता है। यानि सबसे पहले कोड की पंक्ति एक चलाता है, फिर पंक्ति 2, फिर पंक्ति 3, इत्यादि। आगे हम जानेंगे कि कंडीशनल्स की मदद से प्रोग्राम फ़्लो को कैसे नियंत्रित किया जाता है, पर सबसे पहले हमें p5.js के दो मुख्य फंक्शन्स जानने होंगे: `setup()` और `draw()`।



चरण 5: प्रोग्राम के फ़्लो के बारे में जानें (जारी)

	परिभाषा <i>वह क्या है?</i>	विषय-वस्तु <i>इसके अंदर क्या रखा जाता है?</i>
setup()	setup() फंक्शन आपके प्रोग्राम के आरंभ होते समय केवल एक बार चलता है। हर रेखा-चित्र में केवल एक ऐसा फंक्शन होता है और इसे पहली बार के बाद दोबारा कॉल नहीं किया जा सकता है।	ऐसे सभी फंक्शन्स जो आप प्रोग्राम के आरंभ होने पर तुरंत चलवाना चाहते हों, जैसे createCanvas() की मदद से स्क्रीन का आकार, (कभी-कभी) पृष्ठभूमि का रंग, और प्रोग्राम आरंभ होने पर चित्र व फ़ॉन्ट आदि मीडिया लोड करना। यदि आप यहां कोई चर राशि बनाते हैं, तो आप उसे draw() या अन्य फंक्शन्स में प्रयोग नहीं कर सकते हैं।
draw()	draw() फंक्शन अपने ब्लॉक के अंदर लिखी कोड की पंक्तियाँ लगातार तब तक चलाता रहता है जब तक प्रोग्राम रुक नहीं जाता है। यह मुख्य लूप है और यहीं आपके प्रोग्राम का असली कार्य होता है। हर रेखा-चित्र में केवल एक ऐसा फंक्शन होता है और इसे setup() फंक्शन के बाद कॉल किया जाता है।	जो भी चीज़ आप बार-बार करवाना चाहते हों।

इनके बीच के अंतरों को बेहतर ढंग से समझने के लिए, हम यह देखेंगे कि हमारे द्वारा **background()** फंक्शन को रखे जाने के स्थान के आधार पर रेखा-चित्र में किस प्रकार बदलाव होते हैं। यह p5.js कैनवास की पृष्ठभूमि के लिए प्रयुक्त रंग तय करता है। यह कई विभिन्न कलर कीमत पैरामीटर्स स्वीकार सकता है, जिनमें RGB और हैक्साडेसिमल कीमत शामिल हैं। हम रंग के बारे में अगले अनुभाग में और बात करेंगे। फ़िलहाल, हम बस ग्रेस्केल रंग के लिए 0 (ब्लैक) और 255 (वाइट) के बीच का केवल एक कीमत पास करेंगे।

जावास्क्रिप्ट	विवरण
background(redValue, greenValue, blueValue);	<ul style="list-style-type: none"> → background: फंक्शन का नाम। → (): हम कोष्ठकों द्वारा हमारे प्रोग्राम को यह बताते हैं कि उसे इस फंक्शन को कॉल करना है। कभी-कभी हम हमारे कोष्ठकों के अंदर फंक्शन के पैरामीटर या इनपुट शामिल कर देते हैं। → redValue: 0 से 255 के बीच की लाल की कीमत। → blueValue: 0 से 255 के बीच की ब्लू की कीमत। → greenValue: 0 से 255 के बीच की ग्रीन की कीमत। → ;: जावास्क्रिप्ट की सभी कोड की पंक्तियों का अंत सेमीकोलन से होना आवश्यक होता है।

background फंक्शन को रेखा-चित्र के setup() या draw() में रखने से बेहद अलग परिणाम मिलते हैं। इस कोड पर विचार करें:

```
function setup() {
  createCanvas(400, 400);
}

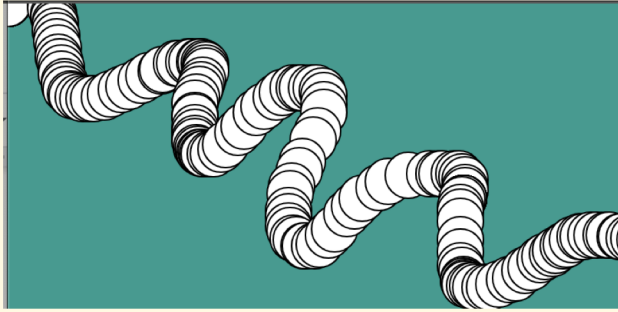
function draw() {
  ellipse(mouseX, mouseY, 50, 50);
}
```

इस समय, इस रेखा-चित्र में कोई पृष्ठभूमि (बैकग्राउंड) नहीं है। यह एक कैनवास बनाता है और स्क्रीन पर जहां माउस कर्सर है वहां एक गोला या दीर्घवृत्त बनाता है। चूंकि ellipse() फंक्शन draw() के अंदर है, अतः हमारा प्रोग्राम अपने हर लूप में स्क्रीन पर एक नया गोला बनाता है, यानी लगभग **60 बार प्रति सेकंड**, और तब तक बनाता रहता है जब तक हम प्रोग्राम को रुकने के लिए नहीं कहते हैं।

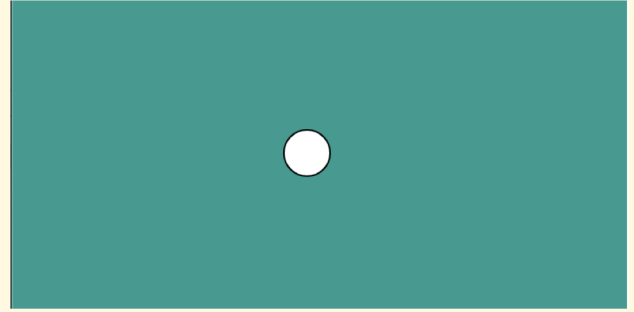
चरण 5: प्रोग्राम के फ़्रेमों के बारे में जानें (जारी)

आइए `draw()` और `setup()` फंक्शन्स के प्रोग्राम फ़्रेमों का अंतर स्पष्ट करने के लिए इन दो उदाहरणों पर नज़र डालते हैं। एक में `background()` फंक्शन `setup()` के अंदर है, और दूसरे में वह `draw()` के अंदर है।

रेखा-चित्र 1



रेखा-चित्र 2



- ❑ **रेखा-चित्र 1** खोलें और अपना माउस कैनवास पर चलाएं। गोला आपके माउस कर्सर के साथ-साथ चलता है और अपने मार्ग पर अन्य गोलों के निशान पीछे छोड़ता जाता है।
- ❑ **रेखा-चित्र 2** खोलें और अपना माउस कैनवास पर चलाएं। अब गोला आपके माउस कर्सर के साथ-साथ चलता है पर निशान नहीं छोड़ता है।
- ❑ **इसके बारे में सोचें:** किस रेखा-चित्र में `background()` फंक्शन, `setup()` के अंदर है? किस रेखा-चित्र में `background()` फंक्शन, `draw()` के अंदर है? क्यों?



अपने विचारों को पृष्ठ 5 पर संदर्भ मार्गदर्शिका से जांचना न भूलें।

चरण 6: समझ की जाँच करें (2 मिनट)

थोड़ा रुकें और p5.js के `setup()` तथा `draw()` फंक्शन्स के संबंध में प्रोग्रामरों की अपनी समझ को जांचें।

आप अपने दोस्त को डिनर में एक बैच पकौड़े, जो उसके मनपसंद हैं, बनाकर उसे चकित करने का मन बनाते हैं। आपको चरण तो याद हैं, पर आप यह सुनिश्चित करना चाहते हैं कि वे सही क्रम में हों। आपको एक पुराना प्रोग्राम मिल जाता है जो आपने पकौड़ों के लिए लिखा था, पर वह अधूरा है!

प्रोग्रामरों के बारे में आपने जो सीखा है उसके आधार पर बताएं कि आप इन क्रियाओं, या “फंक्शन्स” को अपने “प्रोग्राम” में कहाँ रखेंगे ताकि वह ठीक से चले?

वे फंक्शन्स जो आपको जोड़ने हैं:

- रैपर बंद करें
- भरावन मापें
- कड़ाही से पकौड़ा निकालें

वर्तमान प्रोग्राम:

```
setup() {  
  भरावन सामग्रियाँ मिलाएं  
  पकौड़ों के सभी रैपर इकट्ठे करें  
}  
  
draw() {  
  रैपर में चम्मच से भरें  
  पकौड़ा कड़ाही में डालें  
  पकौड़ा तलें  
  पकौड़ा खाएं  
}
```



अपने विचारों को पृष्ठ 6 पर संदर्भ मार्गदर्शिका से जांचना न भूलें।

चरण 7: अपने Girls Who Code at Home परियोजना को साझा करें! (5 मिनट)

हम आपके काम को देखना पसंद करेंगे और हम जानते हैं कि दूसरे भी ऐसा करेंगे। अपना स्यूडोकोड (छद्म कोड) हमसे साझा करें!

[@girlswhocode](#) [#codefromhome](#) को टैग करना मत भूलें, और हो सकता है कि हम आपको हमारे खाते में प्रदर्शित कर देंगे!

और Girls Who Code at Home परियोजनाओं के लिए बनी रहें!

