Prodigy (working title)

**Prodigy is an online massive multiplayer game designed to motivate young students of second languages. It is a powerful tool for teachers to present plain class content in a fun and creative way.**

**Intro**

Say you are a student of psychology (in the real world) and, while it’s nothing to be ashamed of, your English is, less than perfect. So, like the brave do, you decide to enroll to some public English school. In that school, you learn. Or at least you’re supposed to. How do you learn? Depends on you. How are you taught or guided? That depends on the methodology the school follows, but no less important are the people around you – friends and teachers.

You’ve been there for quite some time and made little progress (which is perfectly fine). One day you come across a fellow student. You remember to things about her: she used to be much heavier than she is now, and her English was extremely poor. You chat for a while, and then you notice she’s not to bad (as far as her English goes). You immediately stop her mid speech, and ask:

“What’s your secret?”

“I just practice” she replies.

Now it hits you (you’re a fast learner…), practice is key. You have hardly done any sort of practical exercises. They are an absolute pain the butt (you say to yourself). Well good news, you’re not alone. And don’t beat yourself up about it; most of them are not very fun. And yes, practice absolutely should be.

So why do most practical exercises share this boring nature? One answer – unimaginative content creators. The first common trade great activities share is that they are centered on one main subject learned. Students need to know what it is they are practicing – ambiguity is horrible. The second is presentation. Both are at full control by the teacher. While the subject might be dangerously boring (grammar rules…), it certainly doesn’t have to be presented as such.

**Vision**

This is where the idea of Prodigy comes from. Presenting practical content for students of a second language to practice in a unique environment, separating between content and presentation. Content is controlled by teachers, and until machines speak our language it will always be the case.

An online environment where students practice and learn, engulfed by a story of a young Prodigy on his quest to conquer a field of interest. This is what prodigy is all about. Making study and practice fun. Imagine a virtual universe of many different schools, each focused on some field or skill, not necessarily languages, where players hangout out, learn and solve problems.

Use Prodigy as an in class activity, as presentation of HW, or as an online school. Give students feedback, in real time, create engaging content and activities and put a smile on their faces.

We believe that such an environment will encourage students to learn, even if what they learn is no fun, make a game out of it, do it well, and they shall be exposed to it more. Especially in the first steps one takes learning a new subject, field or skill, first impression is crucial, and determines the rest of their progress.

No such tool exists in the market today.

**The Game of Prodigy (with examples)**

Prodigy is set around the concept of Role Playing Games. A character is a student, the world is the school, and progression is made through accomplishing exercises, named **Quests**.

A Quest, much like an exercise, is basically just a series of questions, aka Challenges. As a unique feature to the game, challenges may be defeated with several steps. In plain English, some questions consist of a number of separate answers that together form a complete solution. For example, if you are required to fill in the blanks in 2 or more different places in a sentence, you are essentially required to form a complete solution from 2 or more separate answers, matching in order of appearance to the blanks in the sentence. So in our Game world, a challenge such as this will require more than one “hit” to be defeated. Alternatively, the Teacher may determine a single minimal score threshold that determines the defeating constraint.

Assume you are the same made up psychology student self from before, and that the actual school where you learn English, used the Prodigy system to create a virtual practical school, designed for its students for the sole purpose of practice. They named it – *Connoisseurs of English.*

As a new comer to the *Connoisseurs of English* class, you first join the first lesson. You are now at **Level** 1. Levels are ordered linearly, and reflect the class curriculum. They are built entirely by a teacher, and have no impact on the rules of the game.

Every Quest, which is an exercise of some X questions, can be completed in various skill levels. A skill level is a number, a percentage to be exact, that describes the amount of correct questions submitted in an entire exercise. For example, accomplishing a Quest on skill level 1 might mean that a player must correctly answer 65% of all questions (or clauses) (We assume for now that the number of different skill levels is static and hardwired into the game engine, meaning the Teacher may modify this mechanism, and is also required to provide values for each skill level.)

Your first **Stage** in the curriculum is made of 3 lessons, and one validation point. It is called *Familiar Grounds*, and is made of three **Zones** representing the three subjects learned – Animals, Numbers and People.

Let’s start with Numbers, should be simple enough. You enter the Numbers Zone. A zone is analogous to a classroom. Teachers and students may be present, some conversing between themselves, others solve some exercises, together or alone. In a zone you are assigned **Missions** you must complete. Completion of all missions opens up the portal to another zone. They are an assortment of **Obstacles** (questions) you must answer correctly, and are the minimal requirements a student must practice in a zone.

Your first Mission is an easy one: “*to count, or not to count?*” and it presents the player with a series of Yes/No questions, with Yes meaning “*to count*”. “*Upon completion you shall receive the following items: Reveal, Flashback*”. Wait what? “*Reveal shows the answer to any obstacle you may encounter. Flashback allows you to return and redo an obstacle during a mission. Use them wisely, as some obstacles prove more challenging than others…*”

Missions are exercises, and provide rewards upon completion. They give instructions, and share a unique format. They define the requirements teachers see as minimal to a reach a fair level of perception in a subject/lesson. Bonus missions may be supplied to inform a player of content regarded as advanced.

Side Note: we shall be concerning ourselves with a small set of such missions, each representing an exercise template (fill-in the blanks, yes/no etc.). At first only closed answer exercises are to be implemented. At a later date, additional types of missions shall be included.

World

Quest 1

Quest nn

**More Points Please!**

Teachers in our school define the thresholds students must pass in order to progress to another zone. This is done using the familiar Experience Points scheme commonly found in RPGs. XPs are granted to a student with every Obstacle he conquers. XPs are always added, never subtracted. This rewarding scheme is crucial to keeping players interested.

The XP scheme is static across any school, lesson and syllabus. Different types of Obstacles may be assigned a different XP scheme, related to the difficulty of the current TYPE of exercise, and not the content. Defining content by levels of difficulty is very abstract and subject to different interpretations by different people. Due to this fact, the progression scheme must be set by the Game rules.

An important feature is the inclusion of items or powers one may acquire during game play. These magical items are special Cheat Tricks one can use. They may help one complete an obstacle or mission, or used as a weapon against other players during a match.

**The Group**

Prodigy is about the individual, fore and foremost.

We do believe however, that some competitiveness may prove helpful. This is accomplished through Group Vs Group play. When one is at a zone, he is actually inside a virtual classroom, able to communicate and view all characters currently in the same zone. From this point, one may throw the glove, and challenge others to a **Rumble**. Players are notified of such an event, and may choose to join. It is limited by max number of concurrent players, and several instances of a Rumble with different players may exist concurrently. In one, groups compete with each other over the highest possible score in a series of Obstacles.

**The Teacher**

Teachers are characters in Prodigy. They are present in the Zones (virtual classrooms) and are available for students at any time. They may participate in games with students, and add content on the fly whenever they please.

Beyond gameplay, teachers follow up on students through the progression of their characters. They may see Obstacles students have had a hard time solving, and gather knowledge of the class as a whole. This is useful beyond comprehension, and eases the work one must usually take to accomplish this.

Teachers are also given a service where exercises may be stored, sorted and fetched at will. They may gain insight on the popularity of some exercises, and improve themselves constantly. This is an extremely powerful tool, usually requesting one to manage all data on his/her own.

**Gameplay**

Gameplay is expressed through the different quests our young prodigy must accomplish. We emphasize again that the content of quests, i.e. the questions in an exercise are created by the World Master. It’s is up to him to introduce engaging, interesting and contextual questions. With the understanding that creating imaginative exercises is not a trivial task for most of us educators, we must augment the usual set of actions and decisions a student has while practicing.

Every challenge in a quest (question) is supplemented with a set of possible answers. This is the main interaction. A player clicks on an answer and receives feedback. Sound shall be used to enhance the experience, indicating a correct answer or a mistake.

The best way to get a grasp the notion of challenges, solutions and answers is to imagine a challenge as an incomplete puzzle, the answers as pieces of the puzzle that are not attached, and the solution as a mapping between an unattached piece and its place in the puzzle. If you fill **all** the missing pieces, you have a solution.

This is a concept not usually found in online exercises. It enables a few things:

1. First, he may be acknowledged for submitting an incomplete solution. If the question requires him to correctly place a minimum of x pieces out of y missing pieces (where x<y), he may pass the question by doing so. This enhances the set of choices a player is confronted (a good thing).
2. Second, more than one solution may be possible per given challenge. This allows more freedom in completing a challenge.
3. Finally, We are able to assign individual weights w (where ) to emphasize certain missing pieces’ importance over others. This allows the teacher to add additional context to an answer.

Features

XP (Intelligence Level)

Player is granted points after every move which results in a positive outcome. For submitting a correct answer, player is granted an amount of points equal (or multiplied by some constant factor) to the weight of the missing piece. For the correct use of any item, the player is rewarded a constant amount of bonus points. Optionally, each item may have its own XP scheme, where the points relevant to the used item are increased, thus allowing upgrades.

Combo (Correct Answer Streak)

Combo is granted ONLY when submitting a correct piece without the use of special items. Combo starts counting when two consecutive pieces have been correctly submitted. The combo reflects implementation of good strategy on player side. If the player selects the next challenge to conquer based on his knowledge (what he knows for sure he does first), than he will be able to gain a large combo, thus charging the items and assisting him with the more difficult challenges to follow.

Items (Special Abilities)

The player can use special items (e.g. health or help) where each use of an item results in either a positive or negative outcome. An item must first be charged before it can be used (this can be achieved by either the combo mechanism, or using some sort of “fuel” to charge items up before or during gameplay.).

Once an item is selected and charged, the outcome depends on the result of the next answer submission. If the submitted answer was correct, the positive outcome of the item is in effect. If the answer was incorrect, the negative outcome is in effect.

Some of the items are:

1. Health: when health is chosen, if the next submitted piece is correct, health is increased, otherwise it decreases player health beyond the normal hit penalty of submitting the wrong answer.
2. Eliminate: when chosen, the player must submit an incorrect answer. If correct, enemy health is decreased by some fair amount (should be calculated relative to the number of incorrect answers available, weight of missing piece etc). If incorrect, combo is lost and user remains in same location.
3. Time Extension: when chosen, the player must submit an incorrect answer. If correct, the clock stops count for X seconds, where X is determined as a constant fraction of the initial clock time. Relevant only for timed quests.
4. Undo: provides undo operations of a previously submitted piece.

Cheats

Cheats are a special king of magic items, but that do not require the player to satisfy some precondition before using them. Once a player has obtained a cheat, he or she may use it whenever they see fit. We shall elaborate later on the means to acquire such cheats. A Cheat always results in a positive outcome, meaning that it results in a correct submission of an answer, either by submitting it, revealing it, skipping it entirely etc. It is however very costly (most possibly in how it is obtained).

Some of the available cheats are:

1. Wild Card: a unique piece that matches any missing piece. It does not reveal the true correct answer to the player.
2. Automatic Elimination: automatically eliminates one (or possibly more) piece that is irrelevant to the current missing piece.

Achievement Levels (Intelligence)

An achievement level is a condition that is evaluated during a quest, which concerns either the percentage of challenges to conquer, or that of pieces to be submitted correctly. There are three (number should not be hardcoded) achievement levels in which a character may complete a puzzle. A quest master must determine values for all three conditions when creating a quest.

**A quest is completed if during play, at least one of the achievement levels is reached.**

For example:

1. 60% = Easy (
2. 80% = Normal
3. 100%+ = Hard

Additional constraints may be imposed in case of timed quests.

Solution Granting

Once a quest has been completed, the character is granted the correct answers (solution) which were submitted by him. Once a solution is in possession, it is available for review purposes at the library (a place where characters may meet, and review passed quests and their solutions).

World Progression

The world is an ordered list of nodes. Characters progress from one node to the next after accomplishing all assignments at a node. A node contains one or more quests, and assignments impose conditions on how to accomplish quests.

Solution Ranking Algorithm

The algorithm acknowledges the order of answers in submitted solutions, calculates the amount of discrepancies between

Let be the current challenge, and be the set of all existing solutions in the challenge.

We define as a vector of strings constructed from the available answers in .

Let be a predefined solution to , where is an integer in the range , and as the solution submitted by the user , where .

We refer to an item in a solution vector as , where is an integer in the range .

Finally, we define to be an item with an empty string.

The goal of the algorithm is to remove items from so that eventually it shall contain a single solution closest to .

The algorithm returns an integer vector ,where , and which holds the following flags:

1. : indicating a correct answer at index j.
2. : indicating an incorrect answer at index j.

We define the following functions:

1. : retains all solutions that contain the item .
2. : retains all solutions that equal the length of the second argument.
3. :

Guy Manzurola